SHAIO CONFERENCE PAPER 2019

Static and Dynamic Input-Output Modelling with Microsoft Excel

Anett Großmann,
Frank Hohmann (GWS)

TABLE OF CONTENTS

FIGURES

ABSTRACT

In the last two decades, the application of Input-output (IO) models has increased due to general availability of extensive structural data sets, esp. IO tables, and the increase in computing power. Today, a broad range of different IO models and applications exists – from simple, static to comprehensive, dynamic IO models.

Simple static IO models are used for comparative-static scenario analysis. With the help of IO quantity models, statements can be made about direct and indirect effects based on exogenous changes in demand. However, these are to be interpreted against the background of their limitations and assumptions such as the time-independency and the non-consideration of feedbacks such as income and price effects. In contrast to the static IO quantity model, the static IO price model is able to capture the effects of input factors such as wages on sectoral production prices. Hereby it is assumed that cost changes are passed on completely and directly. No volume adjustments are made by substitution processes of the customer industries. Both model types can be easily implemented in Microsoft (MS) Excel which provides the necessary functionalities (e. g. matrix algebra functions) without any need for programming tasks. Another advantage is that most IO tables are distributed in MS Excel file formats, e. g. xlsx.

More complex dynamic IO models largely resolve the limitations and inherent assumptions of static IO models. Time is considered explicitly as well as quantity and price reactions are modelled endogenously in a holistic approach and feedback effects are captured. Thus, such models are not only suitable for scenario analysis but also for forecasting. Due to the increased complexity of dynamic IO models, their implementation usually requires extensive programming tasks. Commonly used programming environments are EViews, R or MATLAB.

In this paper, we will show that MS Excel is suitable for the implementation for both static and dynamic IO models. First, we will present a template for a static IO quantity and price model with free available data for Mexico. Second, a template named DIOM-X is discussed which contains all the necessary tools to build a dynamic IO model in MS Excel and its programming language Visual Basic for Application (VBA). The template contains all the necessary tools to perform both scenario analysis and forecasting (data management, regression analysis, model execution engine, presentation of model results). This approach greatly simplifies the task of building a dynamic IO model: Existing knowledge in MS Excel can be used, most computers are already equipped with the software and thus don't any additional software licenses and setup and models may be easily shared or distributed by just transferring one file.

1 INTRODUCTION

In the last two decades, the application of Input-output (IO) models has increased especially due to the increase in computing power and the general availability of extensive structural data sets, esp. IO tables.

Common tasks such as inverting a matrix or solving equation systems which in the past could only be accomplished on expensive workstations or mainframe computers can now be calculated in seconds or a few minutes by cheap desktop or notebook computers. Model builders can now choose from a wide range of programming languages (e. g. Python[1]) and integrated application frameworks (e. g. GAMS, Eviews).

Today's internet technology greatly simplifies the task of collecting the necessary information to build an IO model. In the past, the data had to be extracted either from books or later CD-ROMs. A lot of time-consuming, tedious work was necessary to be able to build the IO database. In recent years, the internet technology greatly simplified the process of data publishing: The number of data sources increased dramatically while the up-date-cycles became much shorter (e. g. the online version of the Eurostat database (https://ec.europa.eu/eurostat/data/database) is updated daily). Furthermore, the number of data formats has been reduced over time which makes data processing less time-consuming. For desktop data processing, most data providers offer either MS Excel, CSV (comma-separated values) or XML (extended markup language) files. In order to be able to use the data in an IO model, it still has to be processed. Most model building tools require the data to be arranged in a certain layout (e. g. values in columns). Even more importantly, variable names (and types) have to be assigned to the data before it can be accessed in the model.

Even with these powerful technologies at hand, the task of building IO models is still a challenge. A potential model builder has not only to get acquainted to IO theory but must have profound knowledge in information technologies (i. e. data processing, programming, spread-sheet programs).

IO models greatly differ in size and possible applications. Simple static IO models are used for comparative-static impact (scenario) analysis. With the help of IO quantity models, statements can be made about direct and indirect effects based on exogenous changes in demand. However, these are to be interpreted against the background of their limitations and assumptions such as the time-independency and the non-consideration of feedbacks such as income and price effects. In contrast to the static IO quantity model, the static IO price model is able to capture the effects of input factors such as wages on sectoral production prices. Hereby it is assumed that cost changes are passed on completely and directly. No volume adjustments are made by substitution processes of the customer industries. Both model types can be easily implemented without any programming tasks. The necessary functionality (i. e. basic algebra, data file format support) is already provided by spreadsheet programs such as MS Excel of OpenOffice. In chapter 2, the implementation of such a model in MS Excel is discussed for the Mexican economy but could also be used for other countries.

[1] Nazara et al. 2003 used Python for IO analysis.

More complex dynamic IO models largely resolve the limitations and inherent assumptions of static IO models. Time is considered explicitly; quantity and price reactions are modelled endogenously in a holistic approach and feedback effects are captured. Thus, such models are not only suitable for scenario analysis but also for forecasting. Due to the increased complexity of dynamic IO models, their implementation usually requires extensive programming tasks. Potential model builders first have to decide on their development environment. They can either build their own development environment by selecting one out of the hundreds of common programming languages (which all have specific strengths and weaknesses) and collecting or implementing programming libraries which carry out specific tasks (e. g. handling of times-series data, matrix algebra). Another option is to get acquainted to a (costly) all-in-one application framework such as GAMS or Eviews. With each of these options, potential model builders will be facing a steep learning curve before they can fully exploit the power of these systems.

Another option to build a dynamic IO model is to use MS Excel not only for data preparation but to build a fully-fledged model in the integrated programming language VBA (Visual Basic for Applications). This approach offers some important advantages:

1. MS Excel is widely used for data processing and evaluation. Thus, most potential model builders are already familiar with at least the basic functionality.
2. Most computers which are used in an professional environment are equipped with the MS Office program suite and are therefore prepared for model development.
3. A model which is fully built in MS Excel can be easily shared or distributed to other users.
4. There is no need for buying other (expensive) model building tools.
5. By reusing existing knowledge in operating MS Excel, the time needed to build a model is greatly reduced which is important for projects with tight time and/or budget restrictions.

In chapter 3, a template for a dynamic IO model and its technical implementation into MS Excel is discussed. Chapter 4 concludes and gives some ideas about further development.

2 STATIC INPUT-OUTPUT MODELS

2.1 OVERVIEW

Standard IO models are able to reveal how different sectors of an economy are interconnected and how changes in one sector affect all other sectors. Their application relies on an IO table providing a comprehensive picture of the supply of goods and services by domestic production and imports, its composition by intermediate consumption and value added as well as the use of goods and services for intermediate and for final consumption, for gross capital formation and exports.

Besides the use of the IO data for descriptive analyses of economic interrelations, this table also provides the empirical fundament for a wide scale of impact analyses of e. g. governmental programs in different fields on domestic production, income and employment. The basis of this kind of analysis is the Leontief equation. Equation (1) shows the reduced form of this equation whereby production *x* is determined by final demand *y* and the Leontief inverse $(I - A_t)^{-1}$ which incorporates the input coefficient matrix *A* and the identity matrix *I* (Eurostat 2008, Leontief 1986).

(1) $x_t = (I - A_t)^{-1} \cdot y_t$

All components of this equations can be derived from an IO table. The resulting set of linear equations (Leontief quantity model or demand pull model) can be used to answer questions such as ‘what happens to the output if final demand due to export promotion or investment changes?’ The results show the impacts on the economy to satisfy an additional (final) demand and give insights into the industry-wide effects (direct and indirect effects).

If the number of employees by industry is also known, the implications for the labour market can be derived in addition to the output effects. Based on equation (1), production-induced employment effects *e* can be computed by multiplying with employment coefficients *b* (equation (2)).

(2) $e_t = b(I - A_t)^{-1} \cdot fd_t$

Another application of the IO table is the Leontief price model (or cost push model, (Miller, Blair 2009, Eurostat 2008, Oosterhaven 1996). It rests on equation (3) where *p’t* is the output price, $(I - A'_t)^{-1}$ is the transposed Leontief inverse, Q is the diagonal matrix with unit factor price for primary inputs and v are the input coefficients for primary inputs.

(3) $p'_t = (I - A'_t)^{-1} Qv$

It can be used to evaluate the impacts of changes in cost factors such as wages, fuel or import prices on output prices by industries.

2.2 IMPLEMENTATION INTO MS EXCEL

The practical implementation of the Leontief demand-pull and cost-push models in Excel is straightforward. In the following subsections the steps to be taken are described in short for both models.

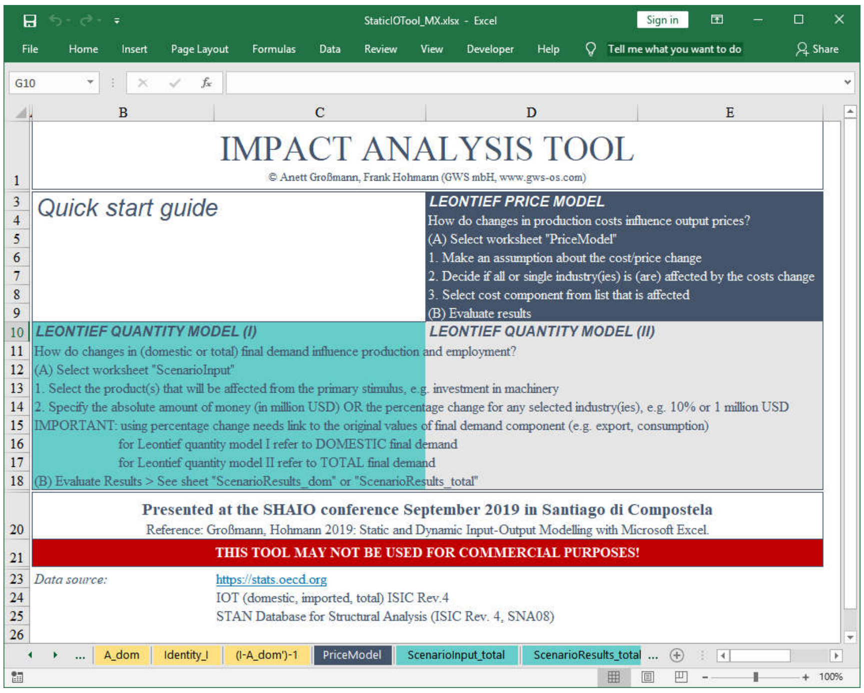
A prerequisite for building a Leontief model is an IO table. For our illustration purposes, we use an IO table for Mexico downloadable from OECD website (https://stats.oecd.org) which includes an IO table for imports, domestic production as well as the total IO table in the ISIC Rev. 4 classification. Employment data are available at STAN Database for Structural Analysis (ISIC Rev. 4, SNA08). These data are the basis for the development of the static open impact analysis tools.

The original data are included in the Excel workbook *StaticIOTool.xlsx*[2] in different worksheets as listed below:

[2] This tool will be made available on request. Please contact the authors.

- *zij_Xj* includes the symmetric IO table in million USD
- *zij_im* includes import matrix in million USD
- *zij_dom* includes domestic production matrix in million USD
- *Labour* includes the employment data in 1,000 persons

All worksheets in the workbook *StaticIOTool.xlsx* with original data are coloured in white. The other worksheets contain the calculation steps to derive the linear equation set for the Leontief quantity model and price model. The scenario input form and results sheet for the price model is coloured in dark blue, for the quantity model in turquoise (Figure 1).



IMPACT ANALYSIS TOOL

© Anett Großmann, Frank Hohmann (GWS mbH, www.gws-os.com)

Quick start guide

LEONTIEF PRICE MODEL
How do changes in production costs influence output prices?
(A) Select worksheet "PriceModel"
1. Make an assumption about the cost/price change
2. Decide if all or single industry(ies) is (are) affected by the costs change
3. Select cost component from list that is affected
(B) Evaluate results

LEONTIEF QUANTITY MODEL (I)   LEONTIEF QUANTITY MODEL (II)
How do changes in (domestic or total) final demand influence production and employment?
(A) Select worksheet "ScenarioInput"
1. Select the product(s) that will be affected from the primary stimulus, e.g. investment in machinery
2. Specify the absolute amount of money (in million USD) OR the percentage change for any selected industry(ies), e.g. 10% or 1 million USD
IMPORTANT: using percentage change needs link to the original values of final demand component (e.g. export, consumption)
for Leontief quantity model I refer to DOMESTIC final demand
for Leontief quantity model II refer to TOTAL final demand
(B) Evaluate Results > See sheet "ScenarioResults_dom" or "ScenarioResults_total"

Presented at the SHAIO conference September 2019 in Santiago di Compostela
Reference: Großmann, Hohmann 2019: Static and Dynamic Input-Output Modelling with Microsoft Excel.

THIS TOOL MAY NOT BE USED FOR COMMERCIAL PURPOSES!

Data source: https://stats.oecd.org
IOT (domestic, imported, total) ISIC Rev.4
STAN Database for Structural Analysis (ISIC Rev. 4, SNA08)

Figure 1: Static Impact Analysis Tool

2.2.1 LEONTIEF QUANTITY MODEL

The following steps have to be taken to build the open Leontief quantity model:

(1) Calculation of input coefficients

(2) Calculation of Leontief-Inverse

(3) Calculation of labour coefficients

(4) Prepare scenario input form

(5) Prepare scenario result sheet

The following explanations only refer to the implementation in MS Excel and do not explain IO theory and the derivation of the Leontief equations (for this please refer e. g. to Leontief 1986, Miller, Blair 2009).

Excel offers helpful functions like *array formulas*, which make cell by cell links superfluous. Furthermore, the *name manager* offers possibilities to define names for ranges, e. g. intermediate demand matrix that can be used for further calculations. Named ranges are much easier to understand and maintain as Excel calculations can often be written just like the formulas in a text book.

In the first step the input coefficients for domestic intermediates *Aij_dom* need to be calculated by dividing domestic intermediate inputs *zij_dom* by total output *xj*. Instead of applying the division in each cells of the *A*-matrix, the use of the name manager and array formulas are suggested.

To define a name, go to the Excel *menu bar*, select Formulas *tab* and click *Define Name*. A new dialog window shows up. In the field *Name* enter a name, e. g. *zij_dom* for the variable. Then select the field *Refers to* and mark the associated cells in sheet *zij_dom*. Finally, click the OK button (Figure 2). After finishing these steps, the name should appear in the name manager.

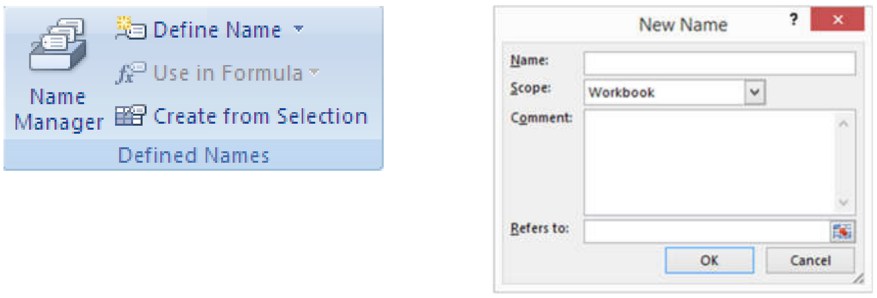
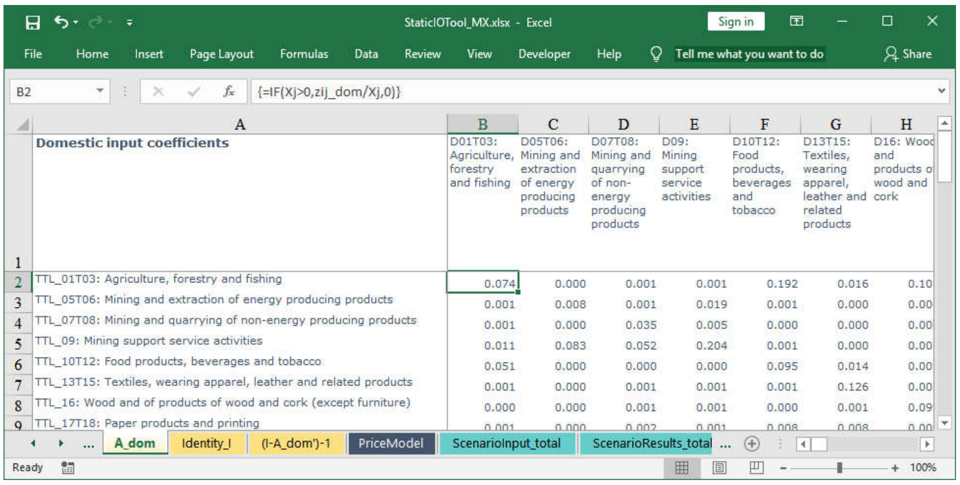


Figure 2: **MS Excel: name manager**

After defining the names for the variables used in the formula, the domestic input coefficients are calculated using an *array formula*. For this, select all cells where the formula should be applied to. In the formula bar the equation – using the defined names – is inserted. Then, press Ctrl + Shift + Enter. The formula is enclosed by curly brackets {} and for the selected dimension the values are calculated according to the formula (Figure 3).



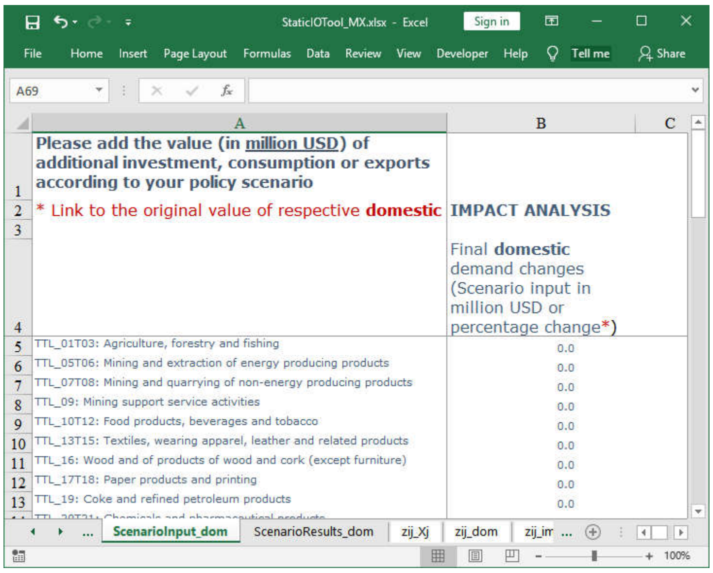
B2 {=IF(Xj>0,zij_dom/Xj,0)}

	A	B	C	D	E	F	G	H
1	Domestic input coefficients	D01T03: Agriculture, forestry and fishing	D05T06: Mining and extraction of energy producing products	D07T08: Mining and quarrying of non-energy producing products	D09: Mining support service activities	D10T12: Food products, beverages and tobacco	D13T15: Textiles, wearing apparel, leather and related products	D16: Wood and products o wood and cork
2	TTL_01T03: Agriculture, forestry and fishing	0.074	0.000	0.001	0.001	0.192	0.016	0.10
3	TTL_05T06: Mining and extraction of energy producing products	0.001	0.008	0.001	0.019	0.001	0.000	0.00
4	TTL_07T08: Mining and quarrying of non-energy producing products	0.001	0.000	0.035	0.005	0.000	0.000	0.00
5	TTL_09: Mining support service activities	0.011	0.083	0.052	0.204	0.001	0.000	0.00
6	TTL_10T12: Food products, beverages and tobacco	0.051	0.000	0.000	0.000	0.095	0.014	0.00
7	TTL_13T15: Textiles, wearing apparel, leather and related products	0.001	0.000	0.001	0.001	0.001	0.126	0.00
8	TTL_16: Wood and of products of wood and cork (except furniture)	0.000	0.000	0.001	0.001	0.000	0.001	0.09
9	TTL_17T18: Paper products and printing	0.001	0.000	0.002	0.001	0.008	0.008	0.00

Figure 3: **Leontief quantity model: calculating input coefficients for domestic intermediates by using defined names and array formula**

In the next step, the Leontief-Inverse is calculated for which an identity matrix *I* needs to be created. MS Excel 2013 and 2016 provide the *MUNIT()*-function that returns an identity matrix for the specified dimension. First, select the dimension of the identity matrix. Then, type the formula =*MUNIT (dimension, e. g. 36)* as an *array formula* (Ctrl + Shift + Enter). A 36x36 matrix with the value one along the diagonal will appear. For matrix inversion, MS Excel provides the *MINVERSE()* function that can be used for calculation of the Leontief equation.

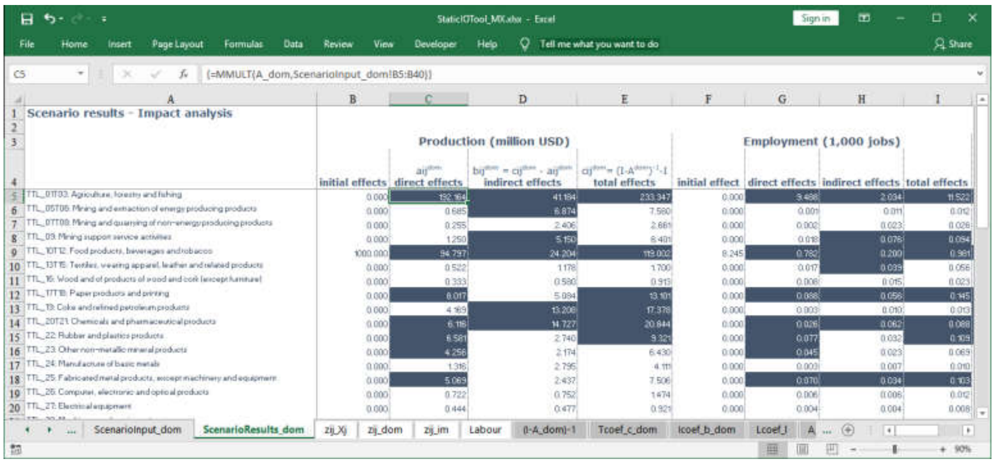
After the derivation of the Leontief equation, the scenario input und result sheet needs to be prepared. Here, the quantified inputs of a scenario (expressed as a change of final demand) will be introduced (Figure 4).



	A	B
1	Please add the value (in <u>million USD</u>) of additional investment, consumption or exports according to your policy scenario	
2	* Link to the original value of respective **domestic**	**IMPACT ANALYSIS**
3		
4		Final **domestic** demand changes (Scenario input in million USD or percentage change*)
5	TTL_01T03: Agriculture, forestry and fishing	0.0
6	TTL_05T06: Mining and extraction of energy producing products	0.0
7	TTL_07T08: Mining and quarrying of non-energy producing products	0.0
8	TTL_09: Mining support service activities	0.0
9	TTL_10T12: Food products, beverages and tobacco	0.0
10	TTL_13T15: Textiles, wearing apparel, leather and related products	0.0
11	TTL_16: Wood and of products of wood and cork (except furniture)	0.0
12	TTL_17T18: Paper products and printing	0.0
13	TTL_19: Coke and refined petroleum products	0.0

Figure 4: **Leontief quantity model: Scenario input sheet**

The scenario input(s) have to be linked to the input and employment coefficients. To calculate the direct, indirect and total input coefficients for production, the inputs from the scenario input sheet are multiplied by the respective coefficient matrix (Figure 5).



{=MMULT(A_dom,ScenarioInput_dom!B5:B40)}

	A	B	C	D	E	F	G	H	I
1	Scenario results - Impact analysis								
2									
3			Production (million USD)				Employment (1,000 jobs)		
4		initial effects	a_{ij}^{dom} direct effects	$b_{ij}^{dom} = c_{ij}^{dom} - a_{ij}^{dom}$ indirect effects	$c_{ij}^{dom} = (I-A^{dom})^{-1}-I$ total effects	initial effect	direct effects	indirect effects	total effects
5	TTL_01T03: Agriculture, forestry and fishing	0.000	192.164	41.184	233.347	0.000	9.488	2.034	11.522
6	TTL_05T06: Mining and extraction of energy producing products	0.000	0.685	6.874	7.560	0.000	0.001	0.011	0.012
7	TTL_07T08: Mining and quarrying of non-energy producing products	0.000	0.255	2.406	2.661	0.000	0.002	0.023	0.026
8	TTL_09: Mining support service activities	0.000	1.250	5.150	6.401	0.000	0.018	0.076	0.094
9	TTL_10T12: Food products, beverages and tobacco	1000.000	94.797	24.204	119.002	8.245	0.782	0.200	0.981
10	TTL_13T15: Textiles, wearing apparel, leather and related products	0.000	0.522	1.178	1.700	0.000	0.017	0.039	0.056
11	TTL_16: Wood and of products of wood and cork (except furniture)	0.000	0.333	0.580	0.913	0.000	0.008	0.015	0.023
12	TTL_17T18: Paper products and printing	0.000	8.017	5.094	13.101	0.000	0.088	0.056	0.145
13	TTL_19: Coke and refined petroleum products	0.000	4.169	13.208	17.378	0.000	0.003	0.010	0.013
14	TTL_20T21: Chemicals and pharmaceutical products	0.000	6.116	14.727	20.844	0.000	0.026	0.062	0.088
15	TTL_22: Rubber and plastics products	0.000	6.581	2.740	9.321	0.000	0.077	0.032	0.109
16	TTL_23: Other non-metallic mineral products	0.000	4.256	2.174	6.430	0.000	0.045	0.023	0.069
17	TTL_24: Manufacture of basic metals	0.000	1.316	2.795	4.111	0.000	0.003	0.007	0.010
18	TTL_25: Fabricated metal products, except machinery and equipment	0.000	5.069	2.437	7.506	0.000	0.070	0.034	0.103
19	TTL_26: Computer, electronic and optical products	0.000	0.722	0.752	1.474	0.000	0.006	0.006	0.012
20	TTL_27: Electrical equipment	0.000	0.444	0.477	0.921	0.000	0.004	0.004	0.008

Figure 5: **Leontief quantity model: scenario results sheet**

The Excel function for performing matrix multiplication is *MMULT*(). The employment effects are calculated by linking the production effects (given in the results sheet) and the labour coefficients.

2.2.2 LEONTIEF PRICE MODEL

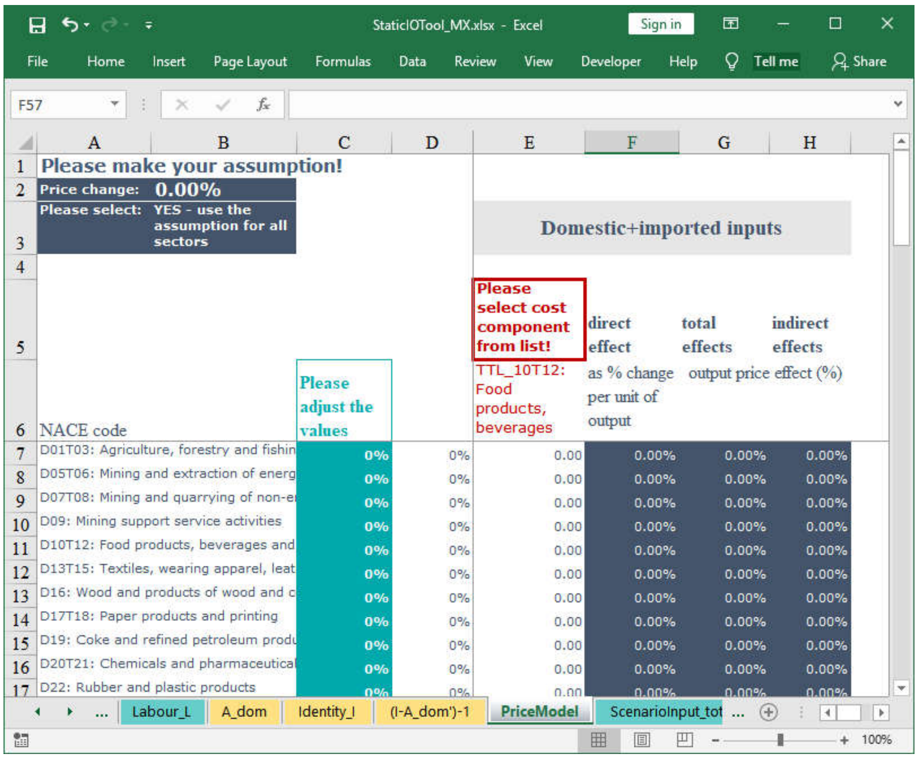
The development of the Leontief price model comprises the following steps:

(1) Calculate input coefficients for domestic intermediates

(2) Calculate price equation $p = (I - A'_{dom})^{-1}$

(3) Prepare price model sheet

The first two steps can be done easily by using the Excel matrix functions *MINVERSE()* and *TRANSPOSE()*. The worksheet "price model" includes the input form for scenario assumptions as well as the results.



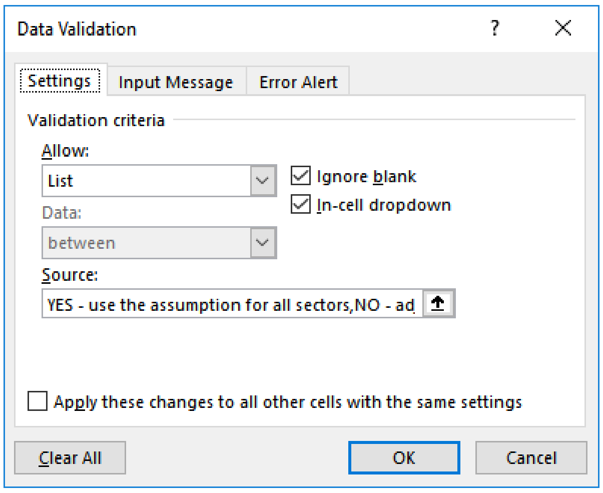
	A	B	C	D	E	F	G	H
1	Please make your assumption!							
2	Price change:	0.00%						
3	Please select:	YES - use the assumption for all sectors			Domestic+imported inputs			
4								
5					Please select cost component from list!	direct effect	total effects	indirect effects
6	NACE code		Please adjust the values		TTL_10T12: Food products, beverages	as % change per unit of output	output price effect (%)	
7	D01T03: Agriculture, forestry and fishin		0%	0%	0.00	0.00%	0.00%	0.00%
8	D05T06: Mining and extraction of energ		0%	0%	0.00	0.00%	0.00%	0.00%
9	D07T08: Mining and quarrying of non-e		0%	0%	0.00	0.00%	0.00%	0.00%
10	D09: Mining support service activities		0%	0%	0.00	0.00%	0.00%	0.00%
11	D10T12: Food products, beverages and		0%	0%	0.00	0.00%	0.00%	0.00%
12	D13T15: Textiles, wearing apparel, leat		0%	0%	0.00	0.00%	0.00%	0.00%
13	D16: Wood and products of wood and c		0%	0%	0.00	0.00%	0.00%	0.00%
14	D17T18: Paper products and printing		0%	0%	0.00	0.00%	0.00%	0.00%
15	D19: Coke and refined petroleum produ		0%	0%	0.00	0.00%	0.00%	0.00%
16	D20T21: Chemicals and pharmaceutical		0%	0%	0.00	0.00%	0.00%	0.00%
17	D22: Rubber and plastic products		0%	0%	0.00	0.00%	0.00%	0.00%

Figure 6: **Leontief price model: input and result sheet (I)**

There are three scenario input cells: In cell B2 the price change assumption is given (Figure 6). In the cell B3 it is specified if the price change assumption is applied for all 36 industries or only for selected industries. In the cell E6 the cost component to which the price change should be applied is given.

In cell B3 a *list* with two entries is created: Either the price assumption is applied to all industries or only for selected industries. The list to be created includes the following two entries: *YES - use the assumption for all sectors, NO - adjust values for each sector individually.* The text is given in the field *Source* (Figure 7).

To create a *list*, first go to the *excel menu bar,* select *data* and click *data validation*. A new dialog window shows up (Figure 7). In the tab '*Settings'* select in the field *Allow*: 'List'. Then, in die field *Source* enter a text or link to the source. Finally, click the OK button.



Validation criteria

Allow: List

Data: between

☑ Ignore blank

☑ In-cell dropdown

Source: YES - use the assumption for all sectors,NO - ad

☐ Apply these changes to all other cells with the same settings

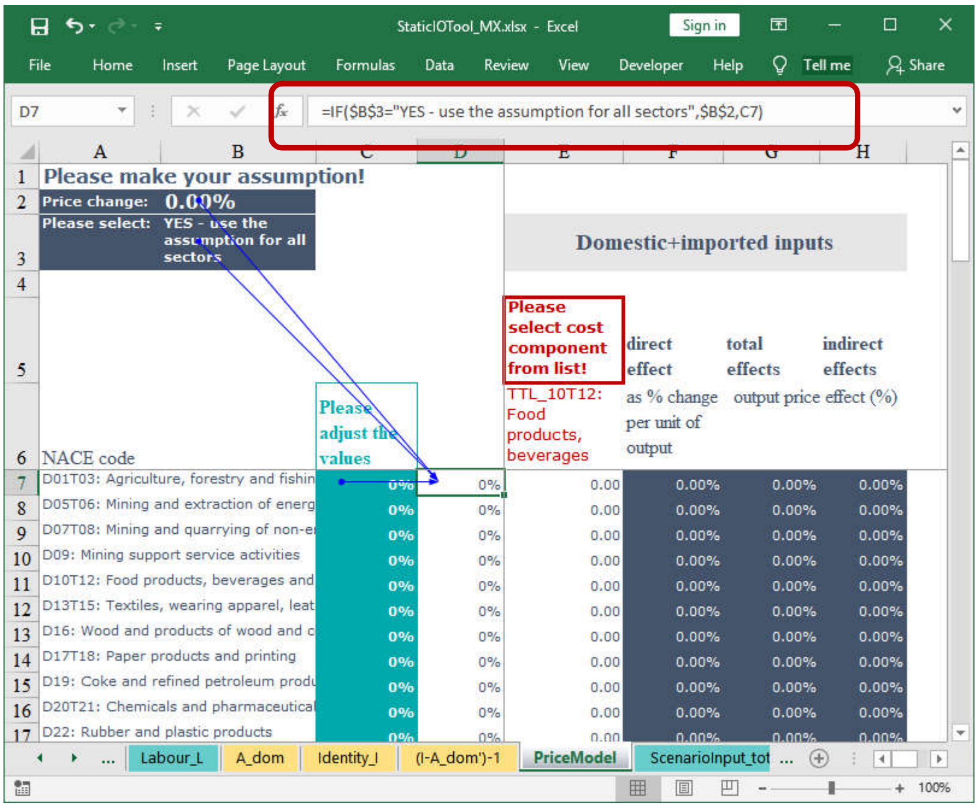
Figure 7: Creating a list in MS Excel

In cell F6 another *list* is created based on the cost components that are given in the domestic flow matrix (sheet zij_dom). Instead of typing each list entry by hand, references to range C9:C48 in sheet zij_dom are used.

The scenario input for all industries is given in cell B2. If the values should be adjusted only for selected industries they have to be given in column C for the suitable industries (rows).

Depending on the YES / NO selection in cell C3 either the value that is given in cell B2 or the values given in column D should be used as an input for the IO price model. The Excel *IF()*-Function is applied in column D to select the appropriate values (Figure 8).

In column E the value of each industry given in row 7 to 42 of a selected cost component in cell E6 is shown. As an example, if the industry 'Food products; beverages' needs input from the industry 'Agriculture, forestry and fishing' then the value given in the domestic flow matrix is linked to cell E11 and multiplied by the assumed price change.

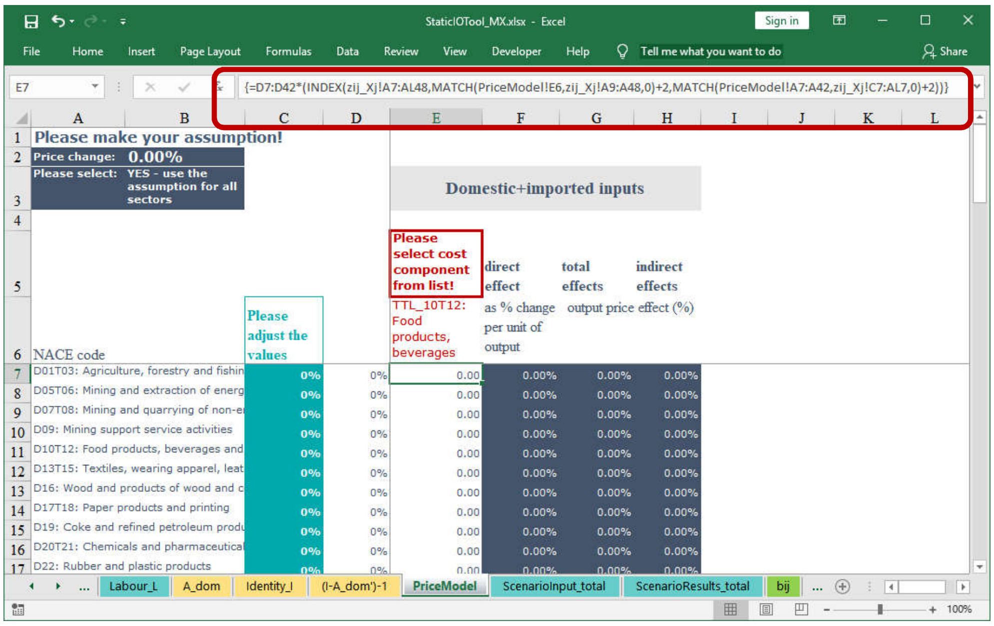


D7 =IF(B3="YES - use the assumption for all sectors",B2,C7)

	A	B	C	D	E	F	G	H
1	Please make your assumption!							
2	Price change:	0.00%						
3	Please select:	YES - use the assumption for all sectors			Domestic+imported inputs			
4								
5					Please select cost component from list!	direct effect	total effects	indirect effects
6	NACE code		Please adjust the values		TTL_10T12: Food products, beverages	as % change per unit of output	output price effect (%)	
7	D01T03: Agriculture, forestry and fishin		0%	0%	0.00	0.00%	0.00%	0.00%
8	D05T06: Mining and extraction of energ		0%	0%	0.00	0.00%	0.00%	0.00%
9	D07T08: Mining and quarrying of non-e		0%	0%	0.00	0.00%	0.00%	0.00%
10	D09: Mining support service activities		0%	0%	0.00	0.00%	0.00%	0.00%
11	D10T12: Food products, beverages and		0%	0%	0.00	0.00%	0.00%	0.00%
12	D13T15: Textiles, wearing apparel, leat		0%	0%	0.00	0.00%	0.00%	0.00%
13	D16: Wood and products of wood and c		0%	0%	0.00	0.00%	0.00%	0.00%
14	D17T18: Paper products and printing		0%	0%	0.00	0.00%	0.00%	0.00%
15	D19: Coke and refined petroleum produ		0%	0%	0.00	0.00%	0.00%	0.00%
16	D20T21: Chemicals and pharmaceutical		0%	0%	0.00	0.00%	0.00%	0.00%
17	D22: Rubber and plastic products		0%	0%	0.00	0.00%	0.00%	0.00%

Figure 8: **Leontief price model: input and result sheets (II)**

Therefore, depending on the selected list entry and the price change assumption the values in range E7:E42 change automatically. The following array formula combining the Excel *INDEX()* and *MATCH()* functions is used:


E7

```
{=D7:D42*(INDEX(zij_Xj!A7:AL48,MATCH(PriceModel!E6,zij_Xj!A9:A48,0)+2,MATCH(PriceModel!A7:A42,zij_Xj!C7:AL7,0)+2))}
```

Please make your assumption!

Price change: 0.00%

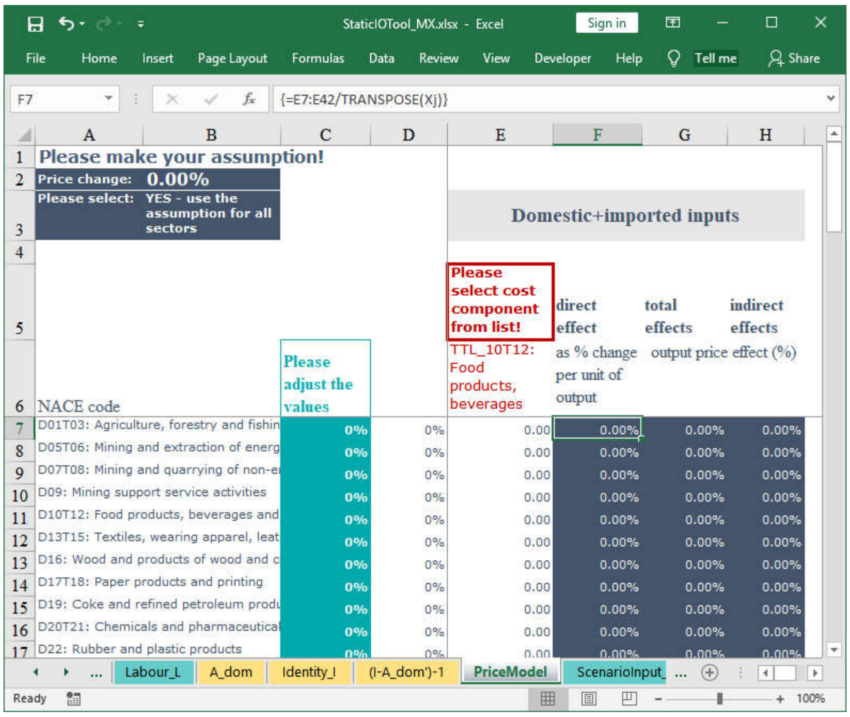
Please select: YES - use the assumption for all sectors

Domestic+imported inputs

NACE code	Please adjust the values		Please select cost component from list! TTL_10T12: Food products, beverages	direct effect as % change per unit of output	total effects output price effect (%)	indirect effects
D01T03: Agriculture, forestry and fishin	0%	0%	0.00	0.00%	0.00%	0.00%
D05T06: Mining and extraction of energ	0%	0%	0.00	0.00%	0.00%	0.00%
D07T08: Mining and quarrying of non-e	0%	0%	0.00	0.00%	0.00%	0.00%
D09: Mining support service activities	0%	0%	0.00	0.00%	0.00%	0.00%
D10T12: Food products, beverages and	0%	0%	0.00	0.00%	0.00%	0.00%
D13T15: Textiles, wearing apparel, leat	0%	0%	0.00	0.00%	0.00%	0.00%
D16: Wood and products of wood and c	0%	0%	0.00	0.00%	0.00%	0.00%
D17T18: Paper products and printing	0%	0%	0.00	0.00%	0.00%	0.00%
D19: Coke and refined petroleum produ	0%	0%	0.00	0.00%	0.00%	0.00%
D20T21: Chemicals and pharmaceutical	0%	0%	0.00	0.00%	0.00%	0.00%
D22: Rubber and plastic products	0%	0%	0.00	0.00%	0.00%	0.00%

Figure 9: Leontief price model: input and result sheets (III)

To calculate the direct output price effect by industry as percentage change per unit of output (column F), entries in column E are divided by the production by industries *Xj* (Figure 10).


F7

```
{=E7:E42/TRANSPOSE(Xj)}
```

Please make your assumption!

Price change: 0.00%

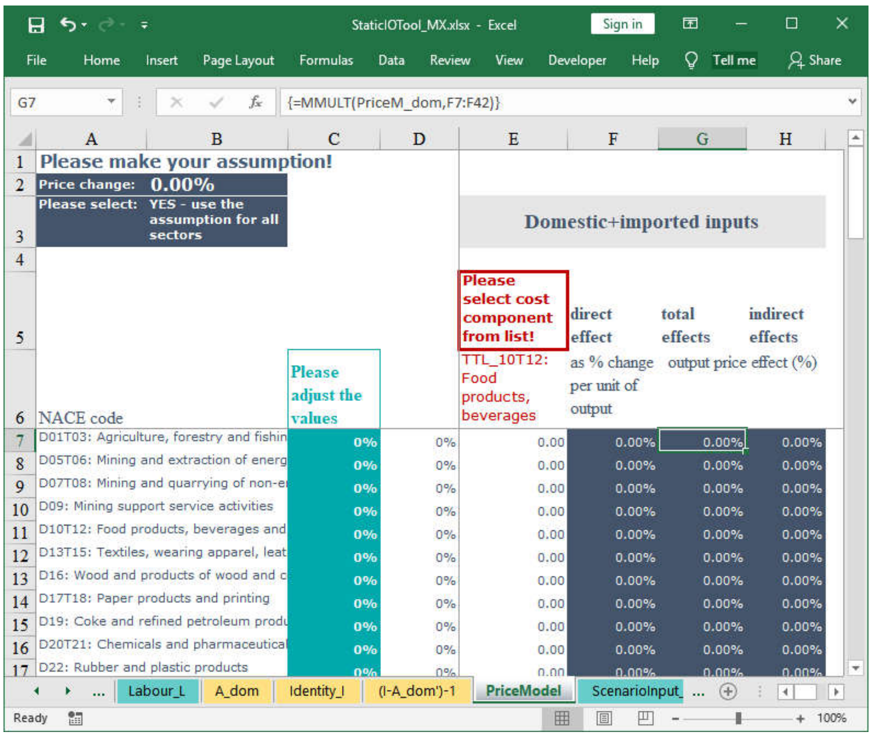
Please select: YES - use the assumption for all sectors

Domestic+imported inputs

NACE code	Please adjust the values		Please select cost component from list! TTL_10T12: Food products, beverages	direct effect as % change per unit of output	total effects output price effect (%)	indirect effects
D01T03: Agriculture, forestry and fishin	0%	0%	0.00	0.00%	0.00%	0.00%
D05T06: Mining and extraction of energ	0%	0%	0.00	0.00%	0.00%	0.00%
D07T08: Mining and quarrying of non-e	0%	0%	0.00	0.00%	0.00%	0.00%
D09: Mining support service activities	0%	0%	0.00	0.00%	0.00%	0.00%
D10T12: Food products, beverages and	0%	0%	0.00	0.00%	0.00%	0.00%
D13T15: Textiles, wearing apparel, leat	0%	0%	0.00	0.00%	0.00%	0.00%
D16: Wood and products of wood and c	0%	0%	0.00	0.00%	0.00%	0.00%
D17T18: Paper products and printing	0%	0%	0.00	0.00%	0.00%	0.00%
D19: Coke and refined petroleum produ	0%	0%	0.00	0.00%	0.00%	0.00%
D20T21: Chemicals and pharmaceutical	0%	0%	0.00	0.00%	0.00%	0.00%
D22: Rubber and plastic products	0%	0%	0.00	0.00%	0.00%	0.00%

Figure 10: Leontief price model: input and result sheets (IV)

The total price effect (column G) is calculated as an array formula by multiplying the price equation ($p = (I - A'_{dom})^{-1}$) and the direct effect given in column F (Figure 11).



G7 {=MMULT(PriceM_dom,F7:F42)}

	A	B	C	D	E	F	G	H
1	Please make your assumption!							
2	Price change:	0.00%						
3	Please select:	YES - use the assumption for all sectors			Domestic+imported inputs			
4								
5					Please select cost component from list!	direct effect	total effects	indirect effects
6	NACE code		Please adjust the values		TTL_10T12: Food products, beverages	as % change per unit of output	output price effect (%)	
7	D01T03: Agriculture, forestry and fishin		0%	0%	0.00	0.00%	0.00%	0.00%
8	D05T06: Mining and extraction of energ		0%	0%	0.00	0.00%	0.00%	0.00%
9	D07T08: Mining and quarrying of non-e		0%	0%	0.00	0.00%	0.00%	0.00%
10	D09: Mining support service activities		0%	0%	0.00	0.00%	0.00%	0.00%
11	D10T12: Food products, beverages and		0%	0%	0.00	0.00%	0.00%	0.00%
12	D13T15: Textiles, wearing apparel, leat		0%	0%	0.00	0.00%	0.00%	0.00%
13	D16: Wood and products of wood and c		0%	0%	0.00	0.00%	0.00%	0.00%
14	D17T18: Paper products and printing		0%	0%	0.00	0.00%	0.00%	0.00%
15	D19: Coke and refined petroleum produ		0%	0%	0.00	0.00%	0.00%	0.00%
16	D20T21: Chemicals and pharmaceutical		0%	0%	0.00	0.00%	0.00%	0.00%
17	D22: Rubber and plastic products		0%	0%	0.00	0.00%	0.00%	0.00%

Figure 11: **Leontief price model: input and result sheets (V)**

The indirect price effect (column H) is calculated by subtracting the direct effect (column F) from the total effect (column G).

2.2.3 SCENARIO ANALYSIS

Once the impact analysis tools are implemented, they can be used for impact analysis which is a technique that analyses the possible consequences (or impacts) of a measure of a future event that is specified in a scenario (e. g. export promotion programs, wage increase). The analysis is comparative-static meaning that a situation before and after an exogenous change is compared but the adjustment process is neglected.

An impact analysis starts with the (1) choice of a policy measure or initial change that should be analysed (scenario design). Then, the appropriate tool (Leontief quantity or Leontief price model) has to be selected. The Leontief quantity model is applied if the direct and indirect impacts of final demand e. g. due to export promotion or investment changes should be evaluated. The Leontief price model is used to evaluate the impacts of changes in cost factors such as wages or fuel prices on output prices by industries.

(2) The policy measure needs to be translated into an impulse – either a change of final demand (Leontief quantity model) or changes in e. g. wages or fuel prices. Furthermore, one or more of the 36 industries that are affected have to be selected.

Afterwards (3), the impacts on production and employment (Leontief quantity model) or output price effects (Leontief price model) are evaluated. In the Excel tools, all results will be updated immediately and automatically.

2.3 CONCLUSIONS

Both model types – the open Leontief quantity model and Leontief price model – can be applied for specific purposes but have their advantages and limitations (see for example, ILO 2017, Großmann et al. 2016, Oosterhaven 1996, Holub, Schnabl 1994).

The advantage of static IO models is that the industry structure and relations are clearly described and transparent, so that the status quo of the value chains, supplier and customer relationships can be analysed comprehensively. Key industries and industries which are vulnerable to exogenous price and volume effects can be identified by applying a comparative-static analysis which show direct and indirect impacts.

Limitations of IO models consist in the assumption that the input structure is fixed (so-called limitational production function) and there are constant returns to scale. There is no substitution of inputs across industries. Future technological changes and innovations are neglected and therefore static IO models might be useful in the short run but cannot be used for analysing structural changes. In the Leontief price model, this restriction (constant input coefficients) implies that changes in costs are passed on completely to downstream industries and they cannot substitute the more expensive input factors by cheaper products. Customers are assumed to be price takers, so that the demand will not change.

Capacity constraints are not considered in the Leontief quantity model. If capacity utilization is high, producers may not be able to satisfy the additional demand in full without investing in e. g. new machinery. Additional demand might also be satisfied with imports. Thus, resulting impacts of a scenario may be overestimated. Furthermore, prices are assumed to be fixed and do not respond to demand shocks. Another aspect – not part of a static open IO analysis – is that the income effect occurring with a higher employment level causes an additional higher demand for consumer goods.

There are many applications of the static IO models. Well-known are for example, IMPLAN (www.implan.com), REMI and RIMS II (BEA (n.d.)) which have some extensions to the very basic static IO model described in this chapter but are still not dynamic IO models which are described in the next chapter.

3 DYNAMIC INPUT-OUTPUT MODEL (DIOM-X)

3.1 OVERVIEW

In contrast to static IO models where the structure of the economy is assumed to be constant, dynamic IO models[3] consider developments over time. The dynamic IO model presented in this chapter is based on the INFORUM approach (Almon 1991, 2014).

As with static IO models, the IO relationships are the focus of the analysis. The models are typically demand-side driven. However, the demand is determined endogenously and not

[3] For examples of dynamic IO models please refer to e. g. Thijs 2017, Eurostat 2008, pp. 527, West 1995, Kratena et al. 2013

given exogenously. The economic cycle is completely represented by the national accounts, so that for the main economic sectors a. o. private households, companies and government, production, income generation and redistribution to consumption can be shown. An important variable in the national accounts (SNAB) is disposable income, which is influenced by both the current labour market situation and the redistributive activities of the government through taxes and subsidies. In addition to other variables such as selling prices, disposable income, it is an important determinant of private consumer demand.

The link between demand and supply is given by the Leontief production function. Furthermore, the IO table shows the cost structure for each industry given by demand for intermediate goods and used primary inputs such as compensation for employees, depreciation, net taxes on production. Prices are derived by using a unit cost approach considering the single cost components. Production prices plus net taxes on goods determine purchasers prices. The latter determine, in addition to disposable income, the demand of private households.

In contrast to simple static IO models, the volume and price reactions in this macro-econometric IO model are empirically based and take the passing on of costs into account and thus include the competitive situation on the different product markets and the labour market.

Supplementary data are population by age groups, employment and wages by industries. Population at working age determines the work force. Labour demand is determined at industry level and related to real production and wages by industries. Increasing real wages tend to lower employment while a higher production level will increase employment. The macroeconomic wage rate is determined by using a Phillips curve approach.

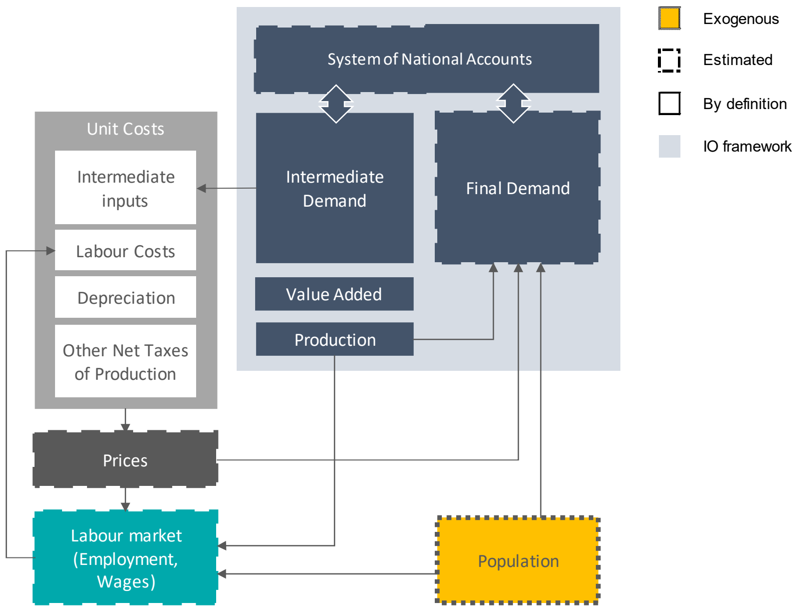


Figure 12: Simplified illustration of a dynamic IO model
Source: own illustration

The model contexts shown in Figure 12 are captured both via identities (e. g. in the IO context) and behavioural equations that are empirically validated. Using econometric methods allows for imperfect markets and bounded rationality (Meyer, Ahlert 2016). However, the specification of the model is not finished with the estimation of single equations. The complete, non-linear, interdependent model equation system is solved iteratively for each year using the Gauss-Seidel algorithm. The iteration process is ended once a given criterion is fulfilled. This criterion has to be an endogenously calculated model variable, e. g. output. As long as the model has not converged, all model equations are recalculated for the current year. Afterwards, all equations are solved for the next years within a given time span. The model template contains the necessary code for the iteration algorithm. A model builder might redefine the criterion.

Exogenous impulses for the model are, for example, population development and exports, which trigger adjustment reactions in the highly interdependent, non-linear model. The modelling approach which covers not only quantity effects but also income and price effects, provides further multipliers that determine the dynamics of the system:

- Leontief multiplier: shows the direct and indirect effects of demand changes (e. g. consumption, investments) on production;
- Employment and income multiplier: Increased production leads to more jobs and thus higher incomes resulting in higher demand (induced effect);
- Investment accelerator: Indicates the necessary investments to maintain the capital stock needed for production based on the demand for goods.

Dynamic IO models exist in different forms and degrees of complexity (see e. g. Eurostat 2008, pp. 527, Stocker et al. 2011, Lehr et al. 2016, Ahlert et al. 2009, Großmann, Lutz 2017, Großmann, Hohmann 2016, Cambridge Econometrics 2014, Lewney et al. 2019). For example, IO models can be modelled bottom-up and top-down: bottom-up indicates that each industry respectively product group is modelled individually and macroeconomic variables are calculated through explicit aggregation. Top-down means that first, the final demand components are determined at the macro level and then are disaggregated accordingly e. g. by using the industry or product shares of the respective variable.

In the following, a template for a simplified dynamic IO model is developed and implemented in MS Excel VBA.

3.2 TECHNICAL FOUNDATION

3.2.1 ESSENTIAL PROGRAMMING LANGUAGE FEATURES

Today, hundreds of programming languages exist and each of them has been developed to overcome some problems or to improve certain features of other languages. In principal, almost any of these programming languages may be used to build an dynamic IO model as long as at least the following features are provided.

A dynamic IO model is processing data over time, thus the language has to support multi-dimensional data structures. These are called *array* in most programming languages. Some languages such as Python do not have a built-in array type. This is not a problem as long as programming libraries are available which offer array-like data structures (for Python, a well-known library for numeric data processing is named “Numpy” (www.numpy.org). In some languages, array-indexing is zero-based (e. g. C, C++) whereas in other languages the first element in an array has the index one. The latter is preferable since most of the published data and sector classifications use one-based indices.

Array-like data structures are not only necessary to process data over time but also to handle data which contains more than one value per year (e. g. sectoral data, IO matrices). For most use-cases, three-dimensional structures are sufficient, i. e. to store a sequence of IO matrices. The speed of indexing into arrays greatly differs between programming languages and is quite important because a comprehensive dynamic IO model contains a considerable amount of these statements. The following table shows the processing time for setting each element in a 10,000x10,000 double-precision floating point array in some popular programming languages[4]. The most simple built-in array implementation in each of the languages was used to receive comparable results.

Interpreted languages are the slowest because instructions are processed one-by-one without much room for optimizations. Compiled code performs fastest because the program as a whole is translated into machine-executable form and usually highly optimized. JIT (Just In Time)-compiled code is often almost as fast as compiled code by translating the program into an intermediate, processor-independent code which is then translated into processor-dependent code at run-time. The table shows that Excel`s built-in VBA language – although it is an interpreted language – performs more than acceptable in comparison to other languages.

Table 1: Processing time for array element processing (10k x 10k elements) in selected programming languages
Source: Own calculations

Language	Time (s)	First Index	Language type
Eviews 9	93	1	Interpreter
Octave 5.1	435	1	Interpreter
R 3.6.1	2628	0	Interpreter
Python 3.6	54	0	Interpreter
Julia 1.0	4.5	1	Interpreter
VBA 2016	***5***	***1***	***Interpreter***
VBScript 5.812	19	0	Interpreter
C# Interactive 3.1	0.5	0	JIT Compiler
Java JDK 1.8	0.2	0	JIT Compiler
C# .NET 4.7	0.6	0	JIT Compiler
MinGW C++ 7.3	0.3	0	Compiler

Additional essential features which are provided by almost every programming language are

- *loops* to be able to iterate over time as well as to address elements in a multidimensional data structure

[4] The size was chosen to simulate a comprehensive model with a lot of indexing statements The calculation was done using an Intel i7-8600k processor with 32GB RAM. The code is available on request.

- *conditional statements* to differentiate cases in order to be able to execute different branches of code
- *sub-programs* (functions, procedures) to create reusable subroutines
- *modules* to divide a program into logical parts

A few programming languages support matrix and vector algebra (e. g. Matlab, Octave) which makes equation statements much more readable. For languages lacking this feature – VBA belongs to this group –, matrix and vector algebra must be implemented by using "functions". The missing functionality is often available in the form of programming libraries which can be downloaded or purchased as add-ons. The model template contains a few additional algebraic functions in order to perform the necessary calculations in the model.

In order to track down errors, a *debugger* is extremely useful: Code may be halted at any statement (often called *break*) as well as data may be evaluated at run-time when the program has been interrupted by a "break" instruction. Some debuggers even allow for evaluating expressions while the program is in interrupted state. MS Excel has a built-in debugger which supports these features.

3.2.2 EXCEL VBA LANGUAGE AND PROGRAMMING ENVIRONMENT

In VBA, (multi-dimensional) arrays may be declared by using the *Dim* or *Redim* statements with the proper bounds given for each dimension[5]. VBA does not directly support matrix/vector algebra but provides functions for some essential tasks such as matrix inversion (*MINVERSE()*) and matrix multiplication (*MMULT()*) which is needed e. g. for the calculation of the Leontief-Inverse. The model template contains additional functions for those that are missing in VBA, e. g. vector addition. In this regard, Excel is not as powerful as other software solutions which might be used for model building.

The variables are empty at program startup and must be filled by reading the data from the worksheets. The model template which is described in section 3.4.4 comes with a simple dataset manager that automatically translates the list of model variables into VBA declarations.

After the model has finished the calculations, the data is transferred back into dataset worksheets. Performing all calculations on arrays is magnitudes faster than reading/writing worksheet cells.

VBA provides the *for* loop statement for a fixed and the *while* loop statement for a varying number of iterations. The *select case* statement is useful if a certain statement out of a group of statements shall be executed. The *if* statement is used when code must be executed with respect to the result of an evaluated expression.

Subroutines may be implemented by using a *sub … end sub* block. If a subroutine has a return value, the statements have to be embraced by a *function … end function* block.

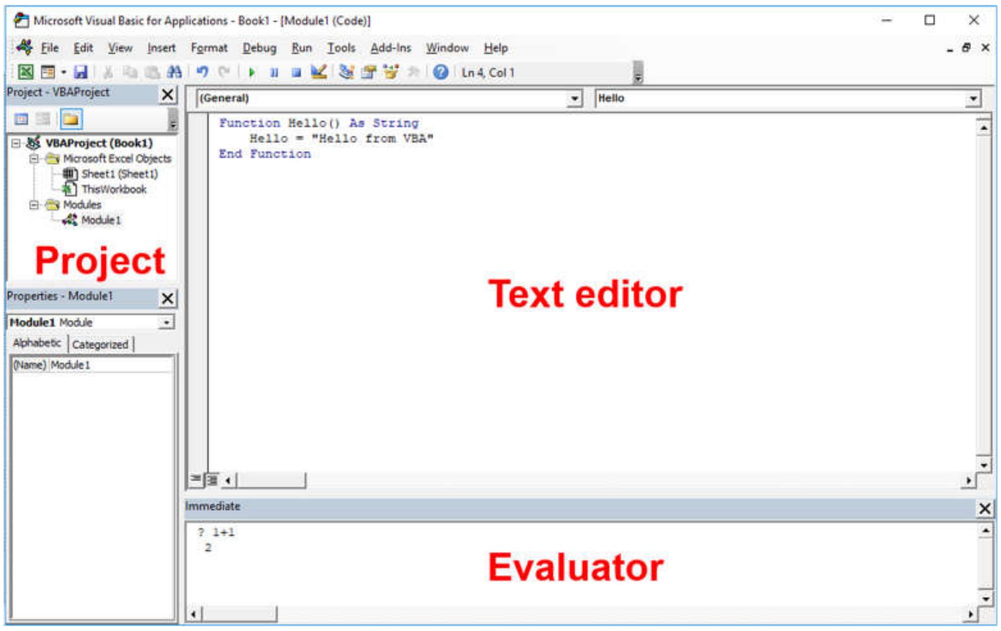
Excel stores all model code along with the data in a *.XLSM (XLS with macros) file or *.XLSB (binary) file. The XLSB format usually offers better compression and loading/saving times. Small to medium size models with some hundred variables may be conveniently stored with

[5] Multi-dimensional arrays are easier to set up but make some algebraic calculations more complicated.

all the code in just one file. The setup and distribution of such a model can therefore be accomplished by just distributing one file to the recipient. With bigger models, it might be more convenient to use separate files, e. g. by separating historical and forecasted data.

By default, the VBA programming environment (Figure 13) is not visible with a default installation of MS Excel or Office, most likely because only the minority of users is using the programming features. The environment can be activated by pressing Alt+F11 or by activating the *Developer* menu entries under *Options, Customize Ribbon*.

The *project* pane gives access to the VBA code which may either be attached to worksheets or be stored in user-created modules. The text editor with syntax highlighting is used to edit the code. The evaluator is helpful for inspecting variables at runtime and for evaluating expressions. By clicking the grey vertical bar next to a certain line in the text editor, a breakpoint may be set in order to suspend the program for error checking, expression evaluation, etc.



```
Function Hello() As String
    Hello = "Hello from VBA"
End Function
```

```
? 1+1
 2
```

Figure 13: MS Excel: VBA programming environment

With the combination of the data stored in worksheets and the VBA programming environment which allows for editing and storing the model code along with the data, a potential model builder has all the necessary tools at hand to start building dynamic IO models.

3.3 STEPS IN MODEL BUILDING

The main steps of building a dynamic IO model are:

1. Database management
2. Regression analysis
3. Core model building
4. Forecasting and scenario analysis

It has to be pointed out that model building is not a sequential process. The model building steps may have to be repeated under certain conditions, e. g. availability of updated or additional data, errors that has been tracked down to previous model building steps or the need for more detailed specifications.

1. Database management

The foundation of every quantitative model is data collected from different sources, i. e. official sources (national and international statistical offices, e. g. Eurostat, OECD.stat). The selection and quantity of data depends on the aim of the modelling exercise.

Data from different sources usually comes in different file formats (e. g. CSV, MS Excel). Even if the file format is the same as with Excel files, data is often organized differently: Time series data may be stored in rows or columns or spread over different sheets. Matrix data may be stored in worksheets or different files (workbooks).

The first important step in building the historical database is to harmonize the data structure by creating a unified dataset where each data item shares the same layout.

One important property missing from original data is the naming of each variable which is needed to access its values from within the model. It is advisable to develop some sort of naming convention which is shared between model builders in order to be able to identify and address each variable in the data set. Additional meta information such as unit, dimension and source should be collected accordingly to further describe the content of the dataset ("meta information"). This information is often useful to avoid errors in certain computational statements such as unit conversions. The model template contains worksheets for both the harmonized dataset and the list of variables with placeholders to fill in this important information.

2. Regression analysis

The availability of historical data is the most important prerequisite for regression analysis which is carried out by econometric models to estimate model parameters for behavioural equations. Instead of using elasticities from the literature, relationships of variables known from e. g. economic theory are econometrically tested against historical data. Regression results (parameters/coefficients) show the direction and magnitude of the relation between the dependent (or left hand side (LHS) variable) and independent variable(s) (or right hand side (RHS) variable(s)). In contrast, variables that are given by definition have a fixed relation to each other.

For example, theory tells that there is a relation between consumption and disposable income. Relative prices and population can play an additional role. Finding the appropriate specification heavily depends on data availability and quality. Furthermore, even with the same specification of the regression equation, the estimated parameters may differ for different countries. That is not really astonishing because people in different countries show different consumption behaviours. Even within one country their behaviour may change over time.

In DIOM-X, regression analysis can be carried out using built-in MS Excel functions such as *RGP()*. Other econometric software such as EViews or R can be used as well. Like with most other model building environments, the data need to be manually transferred to another statistical program and after estimation the parameters must be manually implemented in the model code.

3. Core model building

The main task of this model building step is to create the model structure by putting the set of equations together. Most equations – both regressions and definitions – are not independent but interrelated. LHS variables of one statement occur as RHS variables in another equation. Therefore, the model builder has to carefully define the sequence in which the equations have to be calculated. The model building itself follows the IO approach (Leontief 1986, Miller, Blair 2009). In section 3.4.4, the model template used as an example for a dynamic IO model is introduced.

Furthermore, the model should be divided into logical blocks which then can be implemented (often independently) in the programming language instead of creating a monolithic program structure. The template is organized as a set of modules which interact with each other. It contains modules for the model-independent calculation engine, generated parts such as the variable declarations and model codes.

The template's calculation engine keeps the values constant for those variables for which no computation statements (i.e. regression or definition) have been implemented. This approach makes sure that the values of a such variables cannot drop to zero or are not defined in the forecasting part.

MS Excel's VBA programming language in combination with the pre-structured model template greatly simplifies the task of implementing the model. For example, the calculation of GDP can be written as

v_vacp(t) = v_gosmi(t) + v_coe(t) + v_otlsp(t) with

v_vacp (t):	Value added at current prices
v_gosmi (t):	Gross operating surplus and mixed income
v_coe (t):	Compensation of employees
v_otlsp (t):	Other taxes less subsidies on production

For more please refer to section 3.4.

4. Forecasting and scenario analysis

The DIOM-X model template implements a dynamic IO model, thus all model equations – definitions and behavioural equations – are solved year by year during the simulation period. A forecast usually covers a time span of 10 to 30 years. Due to many feedback linkages the model has no explicit solution and solves all model equations iteratively. The iteration process is ended once a given criterion is fulfilled: For a central model variable, the deviation in percent between two consecutive iterations has to be smaller than a given value (e. g. an output deviation of less than 0.1 % for each industry).

The inherent uncertainty of the future makes it impossible to forecast one 'real' prospective development. The outcome of the most basic forecast is based on the assumption that past behaviour is also effective in the future (business as usual scenario, BAU). Scenario analysis is a method to handle uncertainty and to carry out 'what if'-analysis especially to analyse the impacts of policy measures before implementation. A scenario consists of a set of consistent assumptions which are fed into the model. With DIOM-X, this task becomes extremely easy by typing the desired variable names and values into the *Scenario* worksheet (see section 3.4.4). The alternative scenario is then calculated within a few seconds. Comparing two scenarios reveals differences at a specific point in time and over time (Figure 14) that can be interpreted as reactions to the impulses induced by the initial changes.

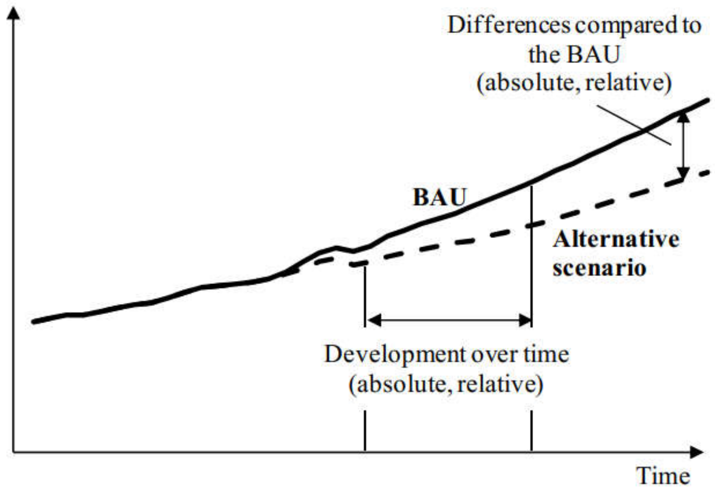

Differences compared to the BAU (absolute, relative)
BAU
Alternative scenario
Development over time (absolute, relative)
Time

Figure 14: **Comparing scenarios**

From a technical point of view, scenarios are adjustments of the value of one model variable or a set of model variables.

A DIOM-X model not only stores the historical data but also the forecasted data which makes it straightforward to visualize the data appropriately (e. g. by creating a management summary graphs and tables for the most important variables).

3.4 IMPLEMENTATION INTO MS EXCEL

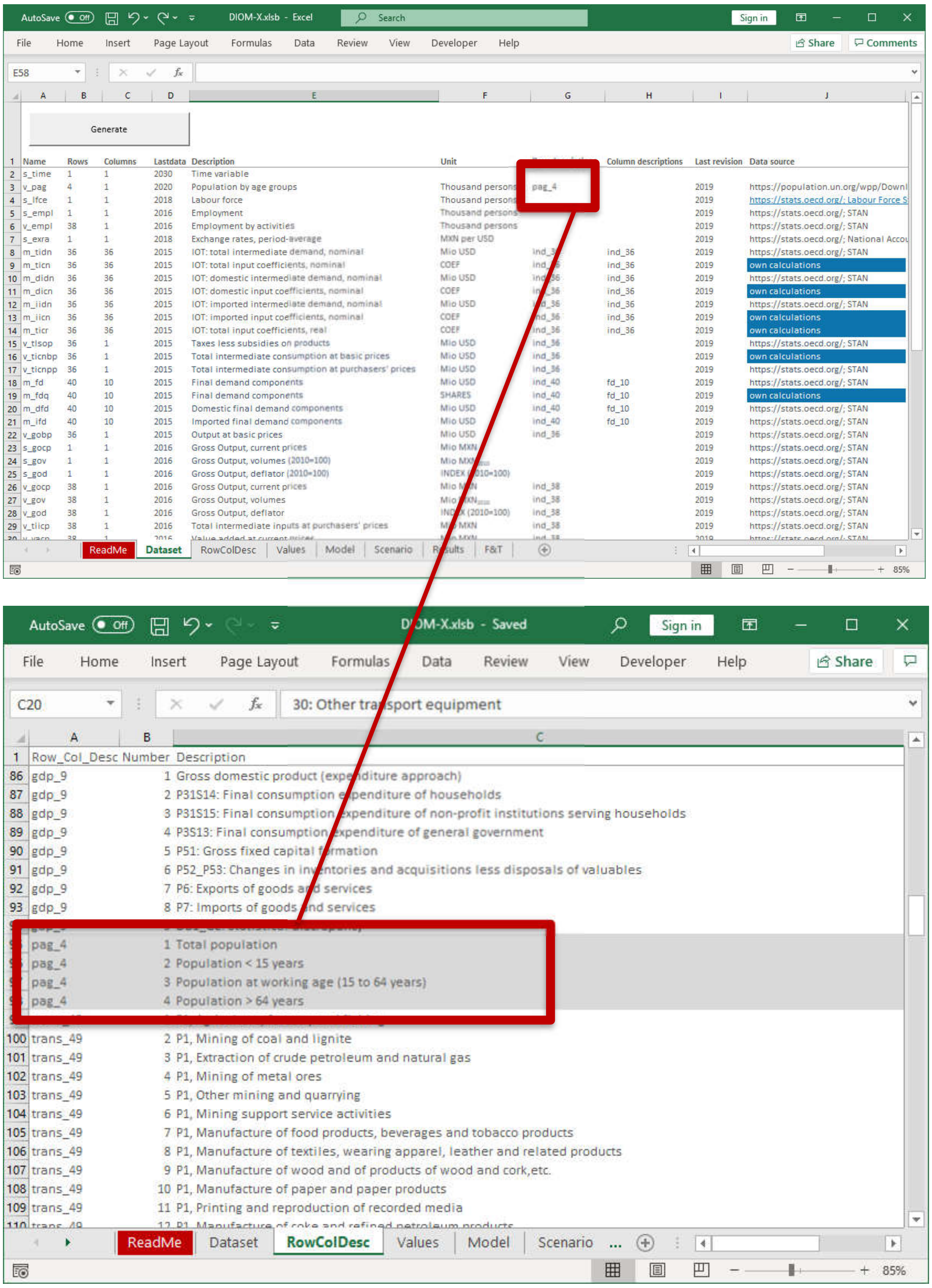
The following sections describe a simple, yet useful template for building a dynamic IO model in MS Excel step-by-step.

3.4.1 DATA MANAGEMENT

In order to retain the usability of an IO model, it often needs to be extended, revised and updated. Its foundation is the historical data set and the quality of the latter has a great impact on the model results. Thus, the data management plays an important role in the life cycle of a model. The time and effort that has to be spend on building the historical database for an elaborated IO model is often underestimated:

- Most data providers offer data in Excel format but the layout of such files is not standardized so that often the data needs to be converted into a format the model is able to handle.
- Data providers from time to time change the layout of their data files which requires that the data processing routines need to be revised.

In order to be able to perform calculations inside the model, each time series has to be given a name. From a model's perspective, the variables with their associated data are sufficient to carry out the calculations. From a modeller's perspective, additional information ("meta data") is needed to keep the model complexity under control, e. g. a description for each variable and its dimensions, units, data source, last revision, etc.



	Name	Rows	Columns	Lastdata	Description	Unit	[illegible]	Column descriptions	Last revision	Data source
2	s_time	1	1	2030	Time variable					
3	v_pag	4	1	2020	Population by age groups	Thousand persons	pag_4		2019	https://population.un.org/wpp/Downl
4	s_lfce	1	1	2018	Labour force	Thousand persons			2019	https://stats.oecd.org/; Labour Force S
5	s_empl	1	1	2016	Employment	Thousand persons			2019	https://stats.oecd.org/; STAN
6	v_empl	38	1	2016	Employment by activities	Thousand persons			2019	https://stats.oecd.org/; STAN
7	s_exra	1	1	2018	Exchange rates, period-average	MXN per USD			2019	https://stats.oecd.org/; National Accou
8	m_tidn	36	36	2015	IOT: total intermediate demand, nominal	Mio USD	ind_36	ind_36	2019	https://stats.oecd.org/; STAN
9	m_ticn	36	36	2015	IOT: total input coefficients, nominal	COEF	ind_36	ind_36	2019	own calculations
10	m_didn	36	36	2015	IOT: domestic intermediate demand, nominal	Mio USD	ind_36	ind_36	2019	https://stats.oecd.org/; STAN
11	m_dicn	36	36	2015	IOT: domestic input coefficients, nominal	COEF	ind_36	ind_36	2019	own calculations
12	m_iidn	36	36	2015	IOT: imported intermediate demand, nominal	Mio USD	ind_36	ind_36	2019	https://stats.oecd.org/; STAN
13	m_iicn	36	36	2015	IOT: imported input coefficients, nominal	COEF	ind_36	ind_36	2019	own calculations
14	m_ticr	36	36	2015	IOT: total input coefficients, real	COEF	ind_36	ind_36	2019	own calculations
15	v_tlsop	36	1	2015	Taxes less subsidies on products	Mio USD	ind_36		2019	https://stats.oecd.org/; STAN
16	v_ticnbp	36	1	2015	Total intermediate consumption at basic prices	Mio USD	ind_36		2019	own calculations
17	v_ticnpp	36	1	2015	Total intermediate consumption at purchasers' prices	Mio USD	ind_36		2019	https://stats.oecd.org/; STAN
18	m_fd	40	10	2015	Final demand components	Mio USD	ind_40	fd_10	2019	https://stats.oecd.org/; STAN
19	m_fdq	40	10	2015	Final demand components	SHARES	ind_40	fd_10	2019	own calculations
20	m_dfd	40	10	2015	Domestic final demand components	Mio USD	ind_40	fd_10	2019	https://stats.oecd.org/; STAN
21	m_ifd	40	10	2015	Imported final demand components	Mio USD	ind_40	fd_10	2019	https://stats.oecd.org/; STAN
22	v_gobp	36	1	2015	Output at basic prices	Mio USD	ind_36		2019	https://stats.oecd.org/; STAN
23	s_gocp	1	1	2016	Gross Output, current prices	Mio MXN			2019	https://stats.oecd.org/; STAN
24	s_gov	1	1	2016	Gross Output, volumes (2010=100)	Mio MXN			2019	https://stats.oecd.org/; STAN
25	s_god	1	1	2016	Gross Output, deflator (2010=100)	INDEX (2010=100)			2019	https://stats.oecd.org/; STAN
26	v_gocp	38	1	2016	Gross Output, current prices	Mio MXN	ind_38		2019	https://stats.oecd.org/; STAN
27	v_gov	38	1	2016	Gross Output, volumes	Mio MXN	ind_38		2019	https://stats.oecd.org/; STAN
28	v_god	38	1	2016	Gross Output, deflator	INDEX (2010=100)	ind_38		2019	https://stats.oecd.org/; STAN
29	v_tiicp	38	1	2016	Total intermediate inputs at purchasers' prices	Mio MXN	ind_38		2019	https://stats.oecd.org/; STAN
30	v_vacp	38	1	2016	Value added at current prices	Mio MXN	ind_38		2019	https://stats.oecd.org/; STAN

C20: 30: Other transport equipment

	Row_Col_Desc	Number	Description
86	gdp_9	1	Gross domestic product (expenditure approach)
87	gdp_9	2	P31S14: Final consumption expenditure of households
88	gdp_9	3	P31S15: Final consumption expenditure of non-profit institutions serving households
89	gdp_9	4	P3S13: Final consumption expenditure of general government
90	gdp_9	5	P51: Gross fixed capital formation
91	gdp_9	6	P52_P53: Changes in inventories and acquisitions less disposals of valuables
92	gdp_9	7	P6: Exports of goods and services
93	gdp_9	8	P7: Imports of goods and services
94	[illegible]	[illegible]	[illegible]
95	pag_4	1	Total population
96	pag_4	2	Population < 15 years
97	pag_4	3	Population at working age (15 to 64 years)
98	pag_4	4	Population > 64 years
99	[illegible]	[illegible]	[illegible]
100	trans_49	2	P1, Mining of coal and lignite
101	trans_49	3	P1, Extraction of crude petroleum and natural gas
102	trans_49	4	P1, Mining of metal ores
103	trans_49	5	P1, Other mining and quarrying
104	trans_49	6	P1, Mining support service activities
105	trans_49	7	P1, Manufacture of food products, beverages and tobacco products
106	trans_49	8	P1, Manufacture of textiles, wearing apparel, leather and related products
107	trans_49	9	P1, Manufacture of wood and of products of wood and cork,etc.
108	trans_49	10	P1, Manufacture of paper and paper products
109	trans_49	11	P1, Printing and reproduction of recorded media
110	trans_49	12	P1, Manufacture of coke and refined petroleum products

Figure 15: Worksheets *Dataset* and *RowColDesc*

While for the model execution only the number of rows and columns is needed, the model builder needs to lookup the description of the elements which are given in worksheet

RowColDesc (Figure 15 above). The entries in column G and H in Figure 15 (top figure) refer to the associate description in worksheet *RowColDesc*.

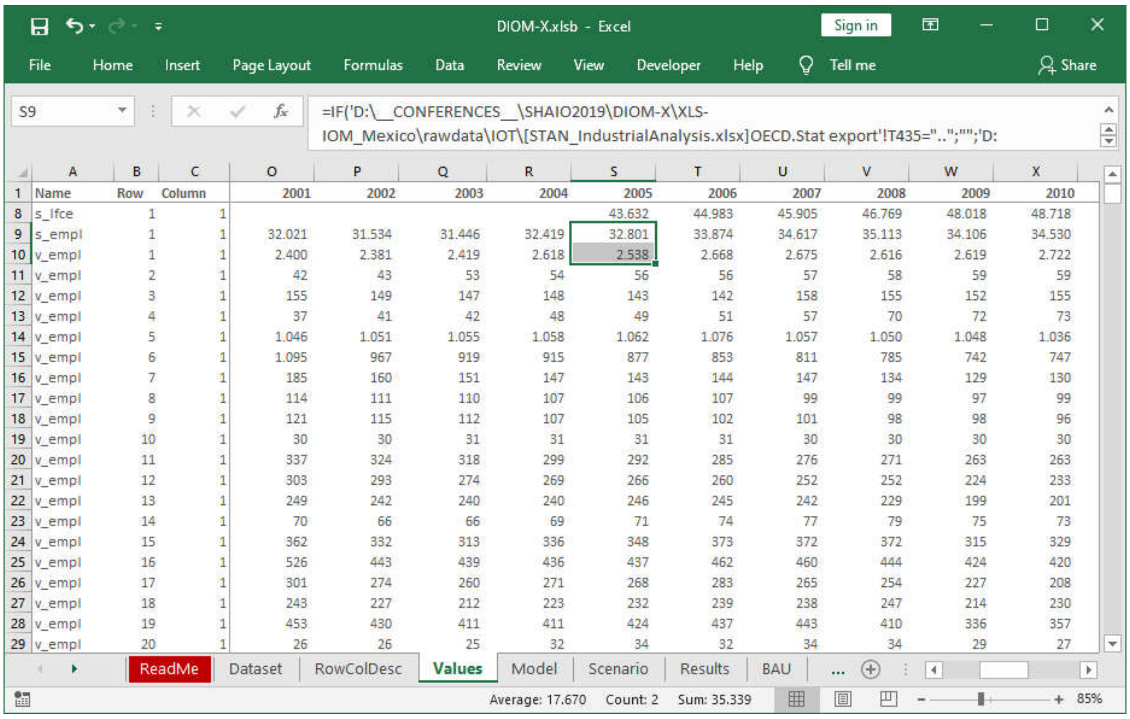
The DIOM-X framework provides the *Dataset* worksheet in which the data contents of the model need to be defined. Each row in the worksheet defines one variable with its basic information such as variable name, last year of available data (lastdata) and the additional meta data such as data source (Figure 15).

By pressing the *Generate* button, the framework automatically translates the full data set (the list of variable definitions) into appropriate VBA variable declaration code.

A model builder should also keep in mind that with a rich dataset, defining variables names becomes a challenge on its own: Very short abbreviations are hard to remember whereas longer, expressive names take much more time to type when it comes to coding. It is recommended to establish some sort of convention which defines a set of rules for naming the variables of the dataset. The convention used for the DIOM-X model template reads as follows:

- Each variable name starts with a letter which describes the variable type followed by an underscore: *s_* means (time) series, *v_* describes a vector and *m_* names a matrix.
- For the name part, the first letter of each noun of the variable description is used, e. g. *fd* means *final demand*.
- The variable names are composed of lower case letters only.

In order to assign the historical values to the variables in the dataset, the template contains the *Values* worksheet (Figure 16):



```
=IF('D:\__CONFERENCES__\SHAIO2019\DIOM-X\XLS-
IOM_Mexico\rawdata\IOT\[STAN_IndustrialAnalysis.xlsx]OECD.Stat export'!T435="..";"";'D:
```

	A	B	C	O	P	Q	R	S	T	U	V	W	X
1	Name	Row	Column	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
8	s_lfce	1	1					43.632	44.983	45.905	46.769	48.018	48.718
9	s_empl	1	1	32.021	31.534	31.446	32.419	32.801	33.874	34.617	35.113	34.106	34.530
10	v_empl	1	1	2.400	2.381	2.419	2.618	2.538	2.668	2.675	2.616	2.619	2.722
11	v_empl	2	1	42	43	53	54	56	56	57	58	59	59
12	v_empl	3	1	155	149	147	148	143	142	158	155	152	155
13	v_empl	4	1	37	41	42	48	49	51	57	70	72	73
14	v_empl	5	1	1.046	1.051	1.055	1.058	1.062	1.076	1.057	1.050	1.048	1.036
15	v_empl	6	1	1.095	967	919	915	877	853	811	785	742	747
16	v_empl	7	1	185	160	151	147	143	144	147	134	129	130
17	v_empl	8	1	114	111	110	107	106	107	99	99	97	99
18	v_empl	9	1	121	115	112	107	105	102	101	98	98	96
19	v_empl	10	1	30	30	31	31	31	31	30	30	30	30
20	v_empl	11	1	337	324	318	299	292	285	276	271	263	263
21	v_empl	12	1	303	293	274	269	266	260	252	252	224	233
22	v_empl	13	1	249	242	240	240	246	245	242	229	199	201
23	v_empl	14	1	70	66	66	69	71	74	77	79	75	73
24	v_empl	15	1	362	332	313	336	348	373	372	372	315	329
25	v_empl	16	1	526	443	439	436	437	462	460	444	424	420
26	v_empl	17	1	301	274	260	271	268	283	265	254	227	208
27	v_empl	18	1	243	227	212	223	232	239	238	247	214	230
28	v_empl	19	1	453	430	411	411	424	437	443	410	336	357
29	v_empl	20	1	26	26	25	32	34	32	34	34	29	27

Figure 16: ***Values* worksheet with data**

For each variable (and in case of vectors and matrices for each element), the model builder needs to the variable name, row and column numbers and values for each year. The model framework reads in this worksheet prior to execution. The worksheet can be created in different ways, depending on the structure and size of the dataset as well as the skills of the model builder:

- Data can be manually copied and pasted from the original data files (not recommended)
- Data may be linked to the original data sources
- The worksheet could be the output of a data processing program
- A mix of the aforementioned options

After successful model execution, the framework automatically copies the full historical dataset as defined in this worksheet as well as the forecasted data to the *Results* worksheet for further processing and evaluation.

3.4.2 REGRESSION ANALYSIS

If the parameters are not taken from the literature, they must be determined on the basis of the country-specific data set. Excel provides the *RGP()* function which does linear regression using the least squares method. Other econometric software such as EViews or R – which provide more statistical techniques – can also be used. The user needs to transfer the dataset from the worksheet *Values* to the program of his choice to perform the estimations.

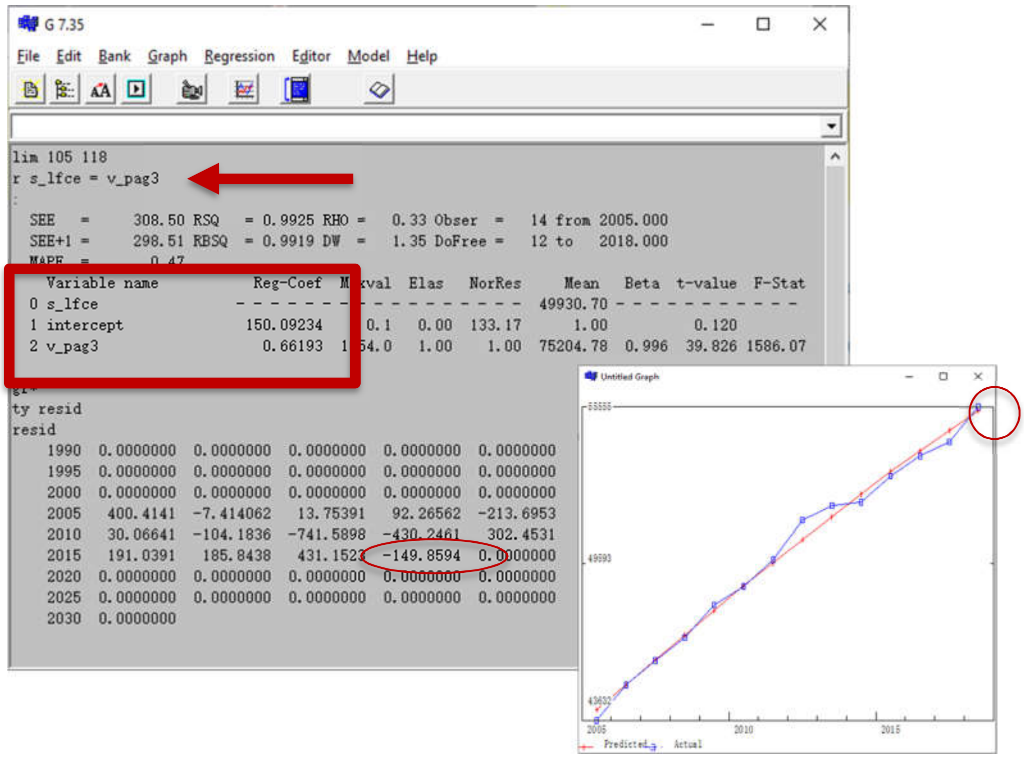
The authors use the econometric regression program G7[6]. In the model template (see Figure 23), the variables are marked with dashed lines, whose parameters are explained econometrically. Among them are macro / time series (e. g. labour force *s_lfce*) as well as vector variables (e. g. GDP components *v_gdpev*).

The following steps describe the procedure of getting macro and vector regressions into the model:

The parameters for the labour force equation are derived by using the OLS technique. At the left hand side of the equation, the variable to be explained is labour force *s_lfce(t)* and on the right hand side is the explanatory variable *v_pag(t)(3)* which is population at working age (see red arrow in Figure 17). Based on the Mexican dataset, the estimation includes historical data from 2005 to 2018.

The parameters (*Reg-Coef*) of the regression equation are shown in Figure 17 (red box) as well. The equation and the parameters must be transferred manually into the VBA code at an appropriate position which is determined by the model logic created by the model builder (Figure 18).

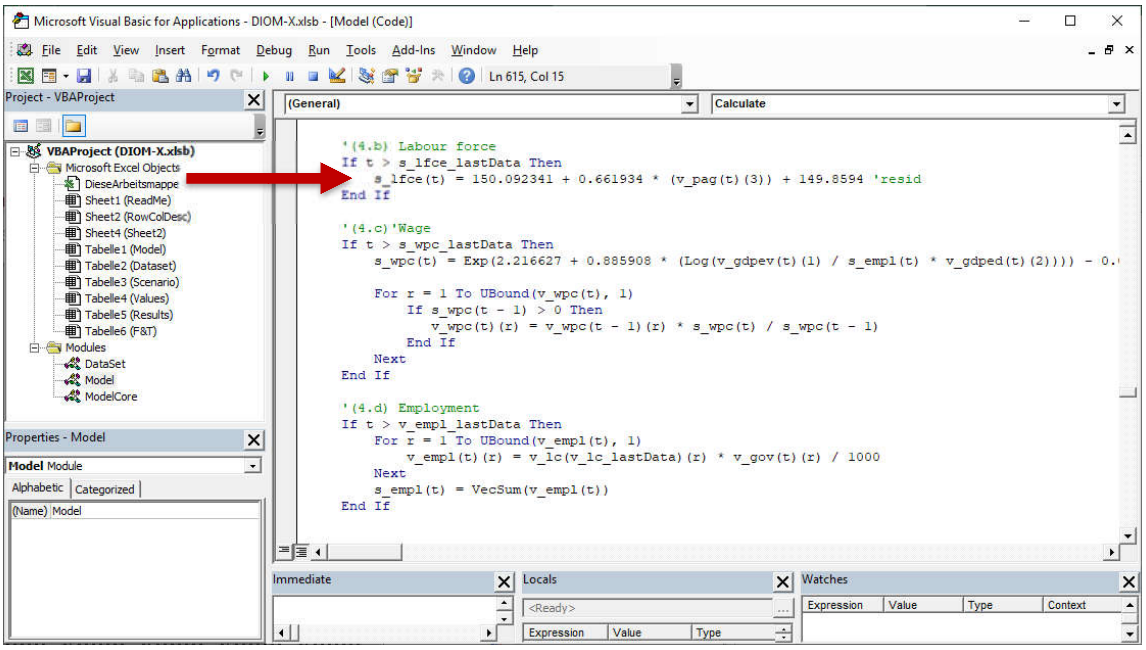
[6] http://www.inforum.umd.edu/software/g7.html



```
lim 105 118
r s_lfce = v_pag3
:
  SEE    =      308.50 RSQ   = 0.9925 RHO =   0.33 Obser  =   14 from 2005.000
  SEE+1  =      298.51 RBSQ  = 0.9919 DW  =   1.35 DoFree =   12 to   2018.000
  MAPE   =        0.47
    Variable name               Reg-Coef  M xval  Elas   NorRes     Mean   Beta  t-value  F-Stat
  0 s_lfce                      - - - - - - - - - - - - - - - -  49930.70 - - - - - - - - - - - -
  1 intercept                  150.09234    0.1   0.00   133.17     1.00           0.120
  2 v_pag3                       0.66193  1 54.0  1.00     1.00  75204.78  0.996  39.826 1586.07

gr*
ty resid
resid
    1990   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
    1995   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
    2000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
    2005    400.4141   -7.414062    13.75391    92.26562   -213.6953
    2010    30.06641   -104.1836   -741.5898   -430.2461    302.4531
    2015    191.0391    185.8438    431.1523   -149.8594   0.0000000
    2020   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
    2025   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
    2030   0.0000000
```

Figure 17: Regression example I

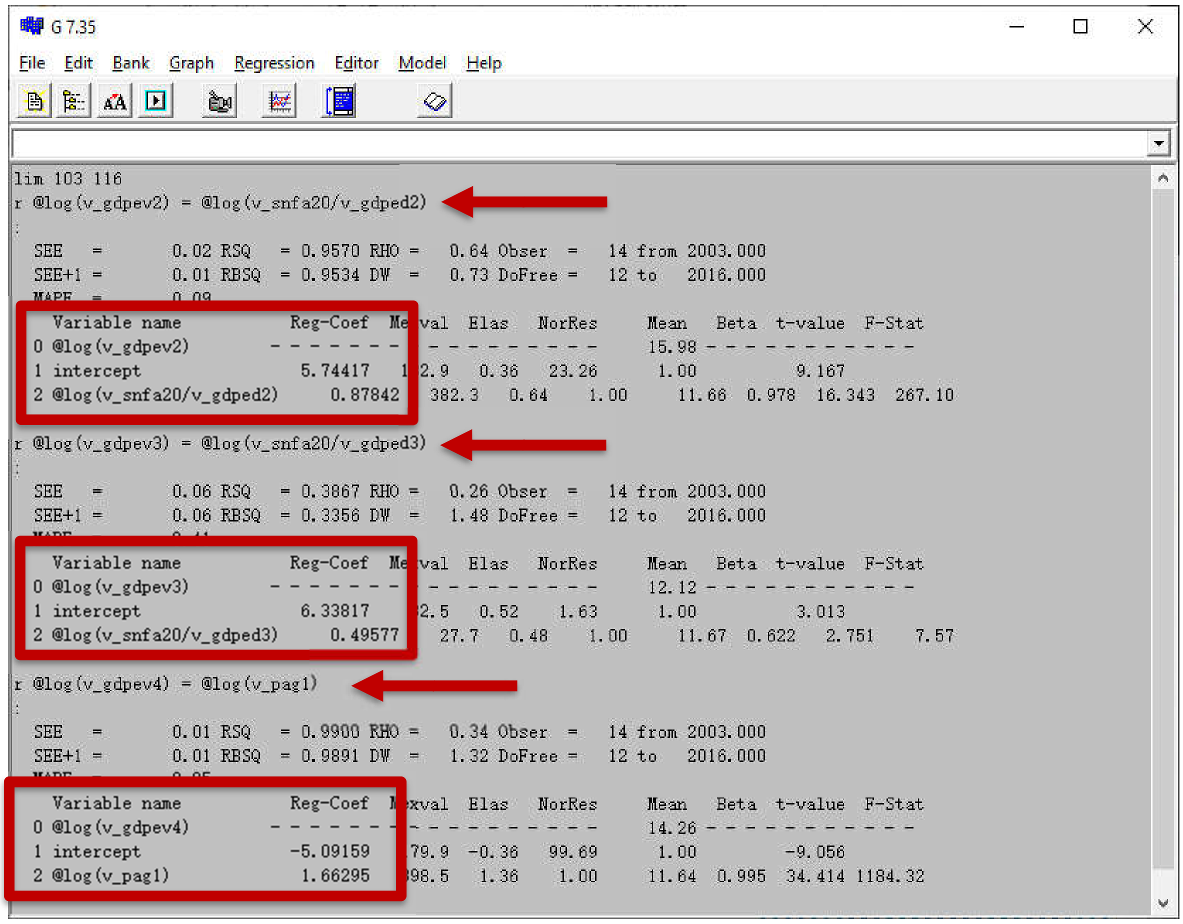
The latter part of the equation is the error term which corrects the estimated value in the last year for which historical data is available. This term ensures a smooth development between the historical and projected data (red circle and graph in Figure 17).



```
'(4.b) Labour force
If t > s_lfce_lastData Then
    s_lfce(t) = 150.092341 + 0.661934 * (v_pag(t)(3)) + 149.8594 'resid
End If

'(4.c)'Wage
If t > s_wpc_lastData Then
    s_wpc(t) = Exp(2.216627 + 0.885908 * (Log(v_gdpev(t)(1) / s_empl(t) * v_gdped(t)(2)))) - 0.

    For r = 1 To UBound(v_wpc(t), 1)
        If s_wpc(t - 1) > 0 Then
            v_wpc(t)(r) = v_wpc(t - 1)(r) * s_wpc(t) / s_wpc(t - 1)
        End If
    Next
End If

'(4.d) Employment
If t > v_empl_lastData Then
    For r = 1 To UBound(v_empl(t), 1)
        v_empl(t)(r) = v_lc(v_lc_lastData)(r) * v_gov(t)(r) / 1000
    Next
    s_empl(t) = VecSum(v_empl(t))
End If
```

Figure 18: Implementing regressions into the VBA code I

A log-log estimation is done to derive price-adjusted GDP components *v_gdpev(t)*. The estimation procedure for series and vector variables differ in one aspect: the row number has to be given (Figure 19).



```
lim 103 116
r @log(v_gdpev2) = @log(v_snfa20/v_gdped2)
:
  SEE   =        0.02 RSQ   = 0.9570 RHO =   0.64 Obser  =   14 from 2003.000
  SEE+1 =        0.01 RBSQ  = 0.9534 DW  =   0.73 DoFree =   12 to   2016.000
  MAPE  =        0.09
    Variable name           Reg-Coef  Me val  Elas   NorRes     Mean   Beta  t-value  F-Stat
  0 @log(v_gdpev2)          - - - - - - - - - - - - - - - -     15.98 - - - - - - - - - - -
  1 intercept                5.74417  1 2.9   0.36   23.26      1.00          9.167
  2 @log(v_snfa20/v_gdped2)   0.87842   382.3   0.64    1.00     11.66  0.978  16.343  267.10

r @log(v_gdpev3) = @log(v_snfa20/v_gdped3)
:
  SEE   =        0.06 RSQ   = 0.3867 RHO =   0.26 Obser  =   14 from 2003.000
  SEE+1 =        0.06 RBSQ  = 0.3356 DW  =   1.48 DoFree =   12 to   2016.000
  MAPE  =        0.41
    Variable name           Reg-Coef  Me val  Elas   NorRes     Mean   Beta  t-value  F-Stat
  0 @log(v_gdpev3)          - - - - - - - - - - - - - - - -     12.12 - - - - - - - - - - -
  1 intercept                6.33817    2.5   0.52    1.63      1.00          3.013
  2 @log(v_snfa20/v_gdped3)   0.49577    27.7   0.48    1.00     11.67  0.622   2.751    7.57

r @log(v_gdpev4) = @log(v_pag1)
:
  SEE   =        0.01 RSQ   = 0.9900 RHO =   0.34 Obser  =   14 from 2003.000
  SEE+1 =        0.01 RBSQ  = 0.9891 DW  =   1.32 DoFree =   12 to   2016.000
  MAPE  =        0.05
    Variable name           Reg-Coef  M xval  Elas   NorRes     Mean   Beta  t-value  F-Stat
  0 @log(v_gdpev4)          - - - - - - - - - - - - - - - -     14.26 - - - - - - - - - - -
  1 intercept               -5.09159   79.9  -0.36   99.69      1.00         -9.056
  2 @log(v_pag1)             1.66295   98.5   1.36    1.00     11.64  0.995  34.414 1184.32
```

Figure 19: Regression example II

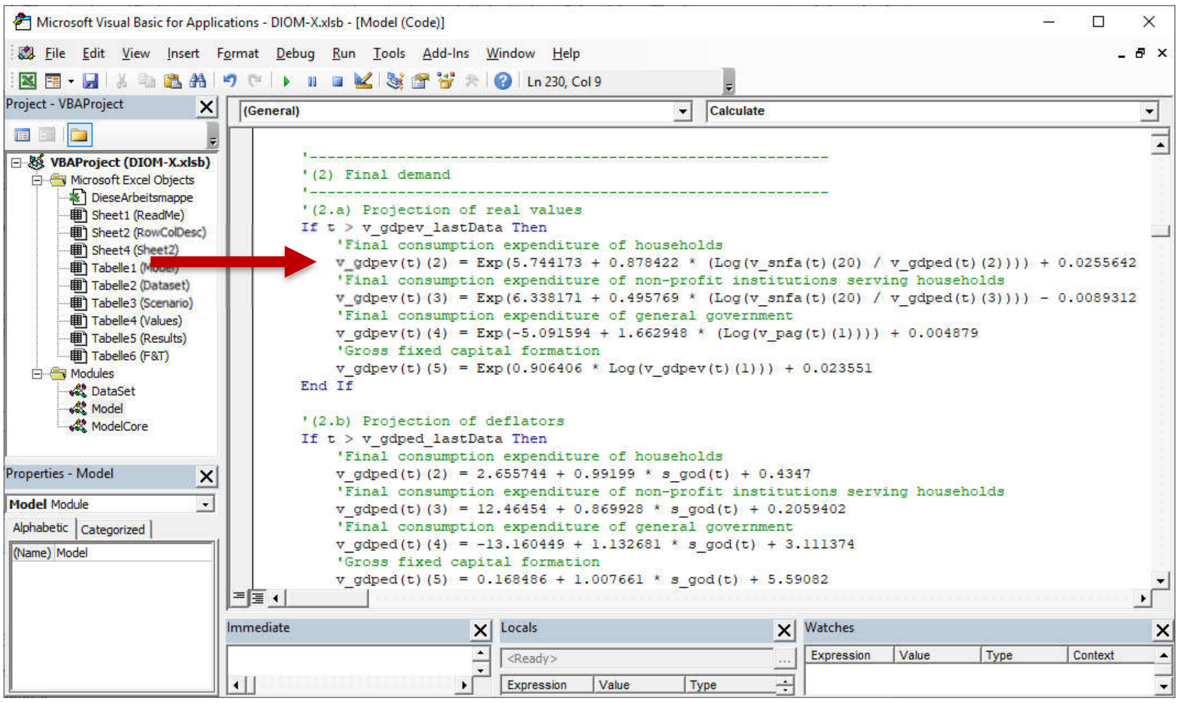
The log-log estimation equation must then be converted into a linear-log equation. The result of the mathematical transformation is defined as:

Log-log: *log(v_gdpev2)=5.744173 + 0.878422*(log(v_snfa20/v_gdped2))+ 0.025*

Linear-log:

v_gdpev(t)(2)=Exp(5.744173+0.878422(Log(v_snfa(t)(20)/v_gdped(t)(2))))+0.025*

Again, the equation and the parameters must be manually transferred into the VBA code at an appropriate position (Figure 20).



```
'-----------------------------------------------------------
'(2) Final demand
'-----------------------------------------------------------
'(2.a) Projection of real values
If t > v_gdpev_lastData Then
    'Final consumption expenditure of households
    v_gdpev(t)(2) = Exp(5.744173 + 0.878422 * (Log(v_snfa(t)(20) / v_gdped(t)(2)))) + 0.0255642
    'Final consumption expenditure of non-profit institutions serving households
    v_gdpev(t)(3) = Exp(6.338171 + 0.495769 * (Log(v_snfa(t)(20) / v_gdped(t)(3)))) - 0.0089312
    'Final consumption expenditure of general government
    v_gdpev(t)(4) = Exp(-5.091594 + 1.662948 * (Log(v_pag(t)(1)))) + 0.004879
    'Gross fixed capital formation
    v_gdpev(t)(5) = Exp(0.906406 * Log(v_gdpev(t)(1))) + 0.023551
End If

'(2.b) Projection of deflators
If t > v_gdped_lastData Then
    'Final consumption expenditure of households
    v_gdped(t)(2) = 2.655744 + 0.99199 * s_god(t) + 0.4347
    'Final consumption expenditure of non-profit institutions serving households
    v_gdped(t)(3) = 12.46454 + 0.869928 * s_god(t) + 0.2059402
    'Final consumption expenditure of general government
    v_gdped(t)(4) = -13.160449 + 1.132681 * s_god(t) + 3.111374
    'Gross fixed capital formation
    v_gdped(t)(5) = 0.168486 + 1.007661 * s_god(t) + 5.59082
```

Figure 20: Implementing regressions into the VBA code II

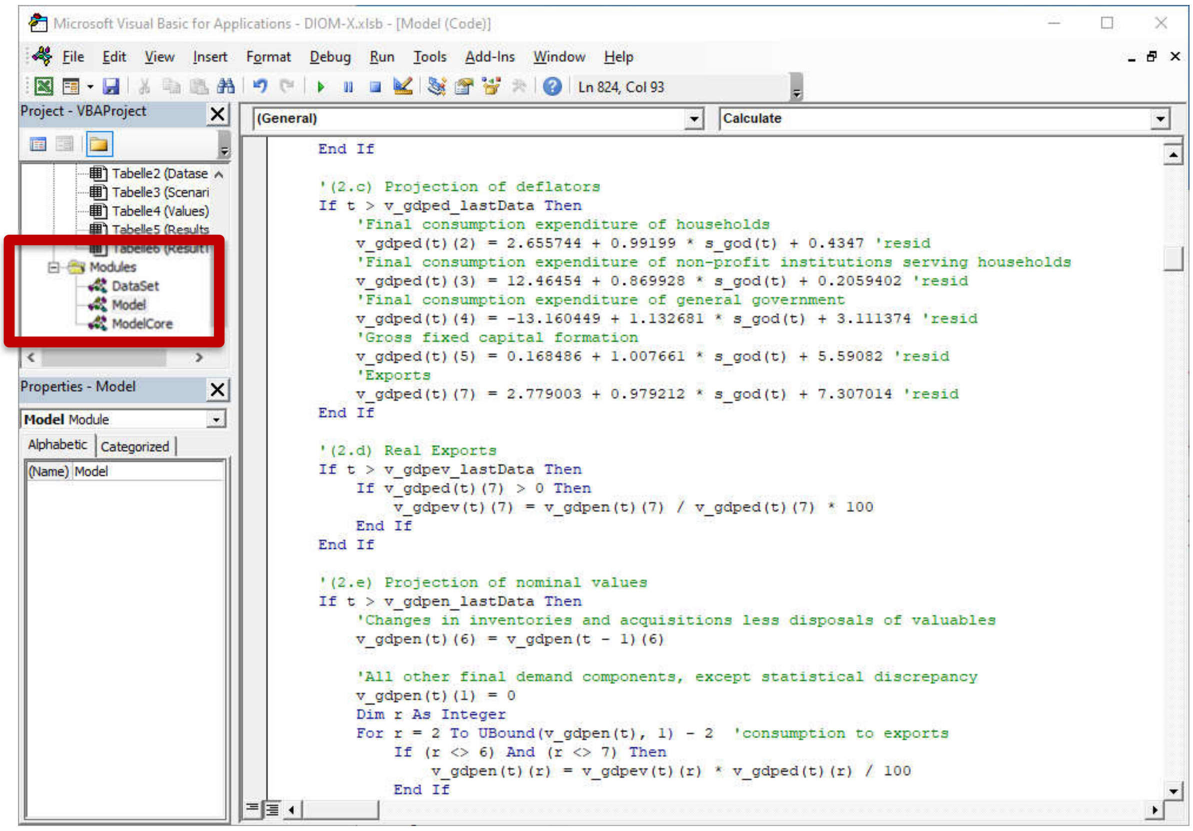
3.4.3 CORE MODEL FRAMEWORK

The DIOM-X model framework is self-contained and a fully stored in the *DIOM-X.XLSB* Excel workbook. It contains three modules (red rectangle in Figure 21):

DataSet: This module contains the model variable declarations which have been automatically generated from the *Dataset* worksheet entries.

ModelCore: The module contains the model-independent source code of the framework that is responsible i. e. for data processing and model execution (see details below). It also contains some (matrix algebra) functions which are not already part of VBA, e. g. vector/matrix sums, addition, subtraction. It also contains the necessary functionality to change variable values at runtime (*tweaks*) which is essential for scenario analysis (see section 3.4.5).

Model: This module contains the user-written source code of the model itself (Figure 21).



```
    End If

    '(2.c) Projection of deflators
    If t > v_gdped_lastData Then
        'Final consumption expenditure of households
        v_gdped(t)(2) = 2.655744 + 0.99199 * s_god(t) + 0.4347 'resid
        'Final consumption expenditure of non-profit institutions serving households
        v_gdped(t)(3) = 12.46454 + 0.869928 * s_god(t) + 0.2059402 'resid
        'Final consumption expenditure of general government
        v_gdped(t)(4) = -13.160449 + 1.132681 * s_god(t) + 3.111374 'resid
        'Gross fixed capital formation
        v_gdped(t)(5) = 0.168486 + 1.007661 * s_god(t) + 5.59082 'resid
        'Exports
        v_gdped(t)(7) = 2.779003 + 0.979212 * s_god(t) + 7.307014 'resid
    End If

    '(2.d) Real Exports
    If t > v_gdpev_lastData Then
        If v_gdped(t)(7) > 0 Then
            v_gdpev(t)(7) = v_gdpen(t)(7) / v_gdped(t)(7) * 100
        End If
    End If

    '(2.e) Projection of nominal values
    If t > v_gdpen_lastData Then
        'Changes in inventories and acquisitions less disposals of valuables
        v_gdpen(t)(6) = v_gdpen(t - 1)(6)

        'All other final demand components, except statistical discrepancy
        v_gdpen(t)(1) = 0
        Dim r As Integer
        For r = 2 To UBound(v_gdpen(t), 1) - 2  'consumption to exports
            If (r <> 6) And (r <> 7) Then
                v_gdpen(t)(r) = v_gdpev(t)(r) * v_gdped(t)(r) / 100
            End If
```

Figure 21: Model source code

The model builder has to provide three subprograms needed by the framework to perform the model calculations:

Initialize: This subprogram may be used to initialize additional variables (e. g. the identity matrix *m_i* for the Leontief inverse).

Calculate: This subprogram contains all the statements which need to be executed at model runtime, i. e. definitions and behavioral equations (see previous screenshot). The DIOM-X framework contains the complete source code for the template model as described in section 3.4.4 of this chapter.

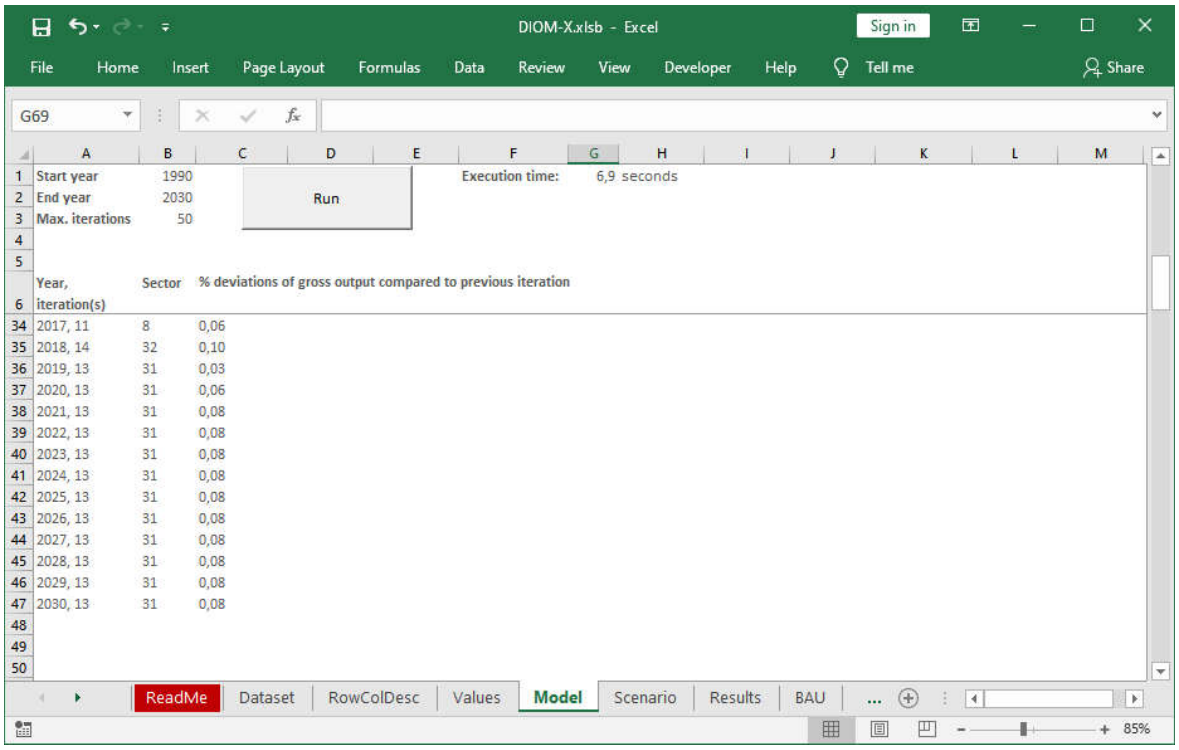
HasConverged: After each iteration, the model framework calls this function in order to detect whether the model has converged in the current year. In a simple model, the function could just return *true* which means that the *Calculate* subprogram should be executed once a year. In a highly interdependent and non-linear equation system using the Gauss-Seidel[7] technique the convergence criteria has to be an endogenously calculated variable. In the Mexican IO model, the very central variable gross output by *v_gov(t)* is used. The model has converged if the percentage deviation in each industry is smaller than 0.1 %.

[7] This method is named after the German mathematicians Gauss and Seidel. They developed an iterative method which is used for solving non-linear systems of equations. This method solves the left hand side of any equation, using previous values for the variables on the right hand side. The computation of a left hand side variable uses the elements of variables that have already been computed. In the next iteration all left hand side variables are calculated again

The number of iterations per year and the industry with the highest deviations but smaller than the convergence criterion is shown in the *Model* worksheet (Figure 22).

To execute the model, the *Run* button on the *Model* worksheet must be pressed (Figure 22). The framework first initializes the model by reading in the historical data as defined in the *Values* worksheet and calling the user-defined *Initialize* subprogram (see above). Next, the frameworks loops through the years by calling the user-defined *Calculate* and *HasConverged* subprograms. Variables which are not explicitly calculated in the calculate subprogram are automatically kept constant for the future in order to avoid calculation errors such as *division by zero*. At the end of the calculation process the full dataset is copied to the *Results* worksheet. If the model could not converge for one year by exceeding the user-defined maximum of iterations, the model will be halted by the framework showing an error message.

The model can also be developed from scratch in order to implement a different modelling philosophy. For such use-cases, an empty "vanilla" version of the model DIOM-X framework has been built which only contains the core framework code. The model can still be executed but does not perform any calculations because the dataset as well as the *Calculate* subprogram are empty. The "vanilla" version may also be used to fill in gaps that has been detected in the historical dataset of a model.



Start year	1990	Execution time:	6,9 seconds
End year	2030		
Max. iterations	50		

Year, iteration(s)	Sector	% deviations of gross output compared to previous iteration
2017, 11	8	0,06
2018, 14	32	0,10
2019, 13	31	0,03
2020, 13	31	0,06
2021, 13	31	0,08
2022, 13	31	0,08
2023, 13	31	0,08
2024, 13	31	0,08
2025, 13	31	0,08
2026, 13	31	0,08
2027, 13	31	0,08
2028, 13	31	0,08
2029, 13	31	0,08
2030, 13	31	0,08

Figure 22: DIOM-X *Model* worksheet

3.4.4 MODEL TEMPLATE AND ITS IMPLEMENTATION IN VBA

This section gives a general overview on the model template and highlights a few statements in the source code as implemented in the subprogram *Calculate*. Commenting the full set of statements is beyond the scope of this paper.

The blueprint of the DIOM-X basic macro-econometric IO model basically follows the INFORUM approach for inter-industry and macroeconomic modelling (Almon 1991, 2014; Meade 2014) which combines IO calculations with econometric methods (West 1995, Lewney 2019).

Figure 23 shows the DIOM-X model template which is based on previous work (Großmann, Hohmann 2013, 2015). The goal was to develop a macro-econometric IO model (or dynamic IO model - DIOM) template based on publicly accessible and constantly updated data sources as well as covering as many countries as possible by using Eurostat and / or OECD datasets. Furthermore, the template serves as a starting point for a more comprehensive model.

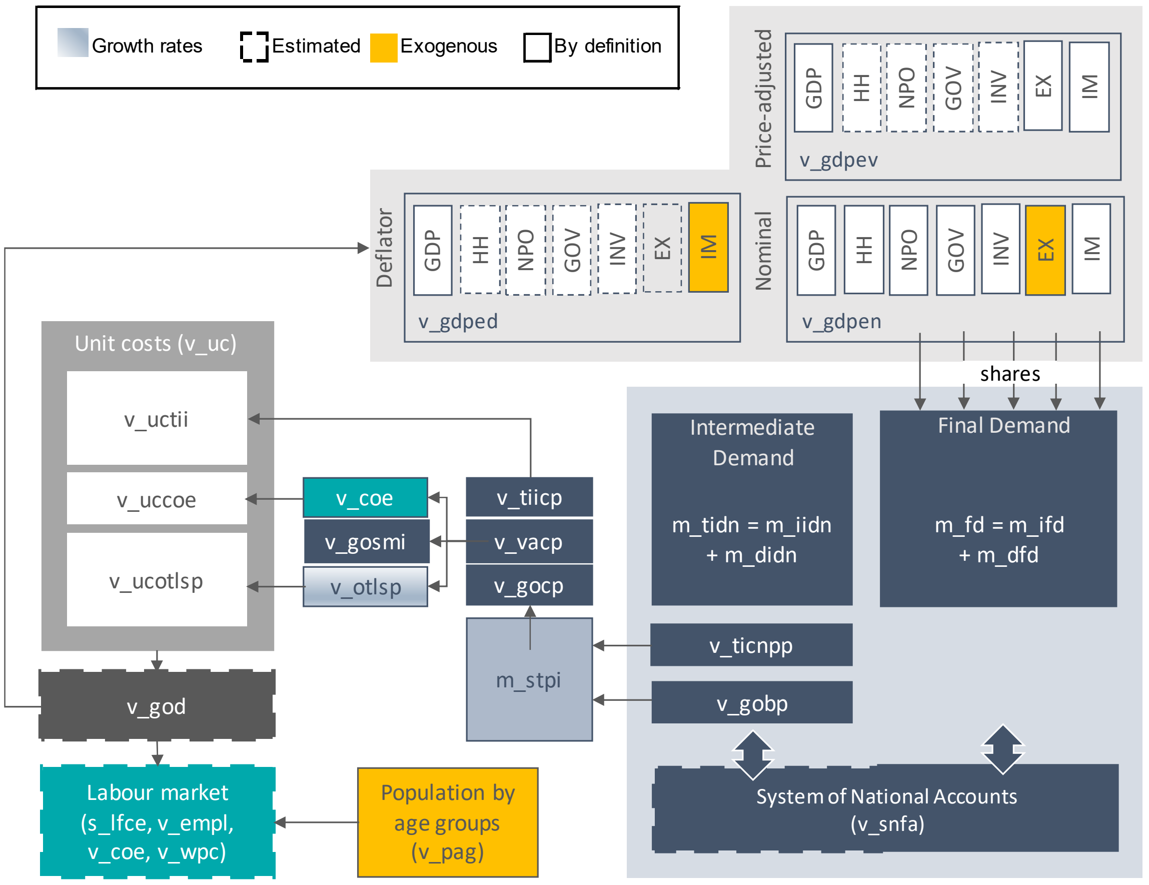


Figure 23: **DIOM-X model template (simplified)**

Economic growth is determined at macro level by modelling quantity and price relationships and then broken down to the industries (top-down approach). The final demand components – either price-adjusted values or nominal values as well as deflators – are determined at macro level: Price-adjusted final consumption expenditures of households *v_gdpev(t)(2)* is computed from real disposable income *v_snfa(t)(20) / v_gdped(t)(2)*. While increasing disposable income positively affects consumption, an overall consumer price increase affects it negatively. Disposable income *v_snfa(t)(20)* is determined in the SNAB. The same formula is used for calculating price-adjusted final consumption expenditures of non-profit institutions serving households *v_gdpev(t)(3)* but the respective deflator is taken (please refer to section 3.4.2).

Price adjusted government expenditures *v_gdpev(t)(4)* depends on the development of total population *v_pag(t)(1)* assuming that the government expenditures increase with population. Gross fixed capital formation *v_gdpev(t)(5)* is a function of price adjusted gross domestic product *v_gdpev(t)(1)*.

The deflators for the GDP components *v_gdped(t)(r)* are estimated with the total output price *s_god(t)* which in turn is calculated by a unit cost approach. *r* indicates the single GDP components starting with GDP (*r=1*) to imports (*r=9*). The import deflator *v_gdped(t)(7*) is usually given exogenously or in the case of Mexico follows the trend. The GDP deflator is calculated by definition as the ratio of nominal GDP and price-adjusted GDP.

Unit costs v*_uc(t)(r)* are defined as costs per unit of price adjusted output by industry *v_gov(t)(r)* while *r* represents all industries (here: 1 to 38 as given from the STAN database) as given from the data. Main cost components are labour costs *v_uccoe(t)(r)* and intermediate costs *v_uctii(t)(r)*. The labour costs by each industry is calculated from the total compensation of employees and the price-adjusted output at industry level. The intermediate costs are calculated from the price-adjusted intermediate inputs *m_ticr(t)(r,c)* purchased by the respective industry at a given price *v_god_s(t)(r)*.

Output deflators *v_god(t)(r)* are estimated by total costs *v_uc(t)(r)* at industry level. The magnitude of mark-up pricing depends on the predominant market structure. In monopolistic markets, the profit margin tends to be higher than under competition.

Nominal GDP components *v_gdpen(t)(r)* are basically derived from deflators *v_gdped(t)(r)* and corresponding price-adjusted values *v_gdpev(t)(r)*. An exception exists for exports *v_gdpen(t)(7)* which are given exogenously.

The transition from the domestic concept (GDP expenditure approach) to the national concept (final demand *m_fd(t)(r,c)*) occurs under consideration of direct purchases abroad by residents and non-residents as well as the exchange rate *s_exra(t)* because the IO tables are given in USD.

In a next step, the total values of final demand are split to the single industries by using constant shares for each final demand component. Additionally, total final demand *m_fd(t)(r,c)* is divided into domestic *m_dfd(t)(r,c)* and imported final demand *m_ifd(t)(r,c)* by applying historically observed import shares by component.

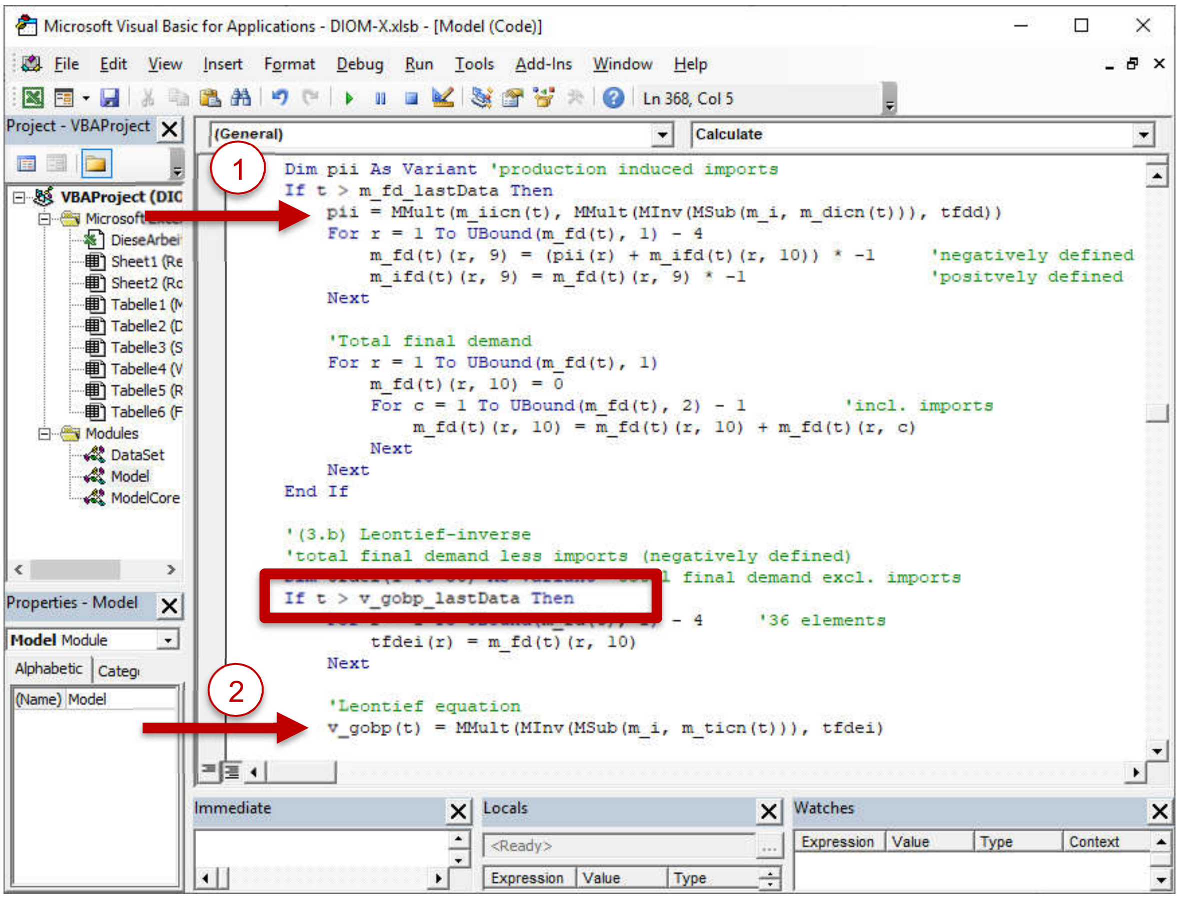
Production-induced imports *pii(t)(r)* – which are based on domestic production – are calculated by $pii = m\_iicn(t) * (m\_i(r,c) - m\_dicn(t))^{-1} * tfdd$

with:	m_iicn(t)	-	Imported input coefficients, nominal
	m_i	-	Identity matrix
	m_dicn(t)	-	Domestic input coefficients, nominal
	tfdd	-	Total final domestic demand

The sum of imported final demand and production-induced imports results in total imports which have to be subtracted from total final demand before entering the Leontief equation $(m\_i, m\_ticn(t))^{-1}$ from which the output by industries *v_gobp(t)(r)* can be derived.

Figure 24 depicts the implementation of the formula for production-induced imports (see arrow 1) and the Leontief equation (see arrow 2) in VBA code which is straightforward by using the Excel functions for matrix multiplication *MMULT* and matrix inversion *MINVERSE*. Additional functions that are missing in VBA such as matrix subtraction *MSub* are implemented in the core model framework (refer to section 3.4.3).

Both formulas are only calculated if the current year *t* is greater then the lastdata (specified in the worksheet *Dataset*) of the respective variable (red rectangle in Figure 24). This condition avoids that the historical values are overwritten.



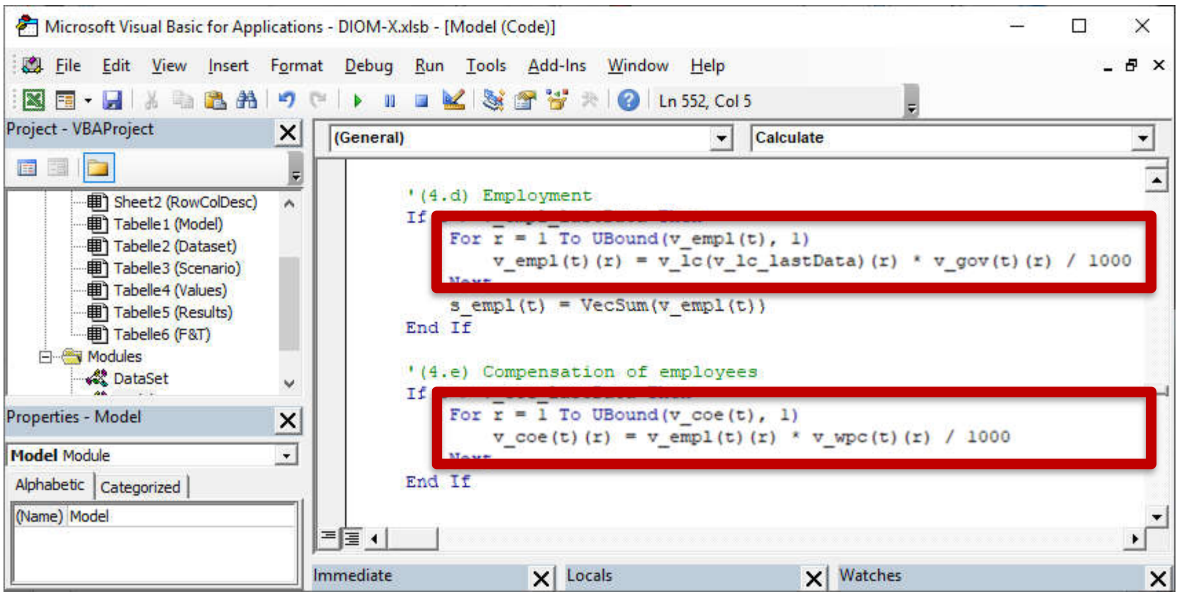
```
Dim pii As Variant 'production induced imports
If t > m_fd_lastData Then
    pii = MMult(m_iicn(t), MMult(MInv(MSub(m_i, m_dicn(t))), tfdd))
    For r = 1 To UBound(m_fd(t), 1) - 4
        m_fd(t)(r, 9) = (pii(r) + m_ifd(t)(r, 10)) * -1      'negatively defined
        m_ifd(t)(r, 9) = m_fd(t)(r, 9) * -1                  'positvely defined
    Next

    'Total final demand
    For r = 1 To UBound(m_fd(t), 1)
        m_fd(t)(r, 10) = 0
        For c = 1 To UBound(m_fd(t), 2) - 1          'incl. imports
            m_fd(t)(r, 10) = m_fd(t)(r, 10) + m_fd(t)(r, c)
        Next
    Next
End If

'(3.b) Leontief-inverse
'total final demand less imports (negatively defined)
Dim tfdei(1 To 36) As Variant 'total final demand excl. imports
If t > v_gobp_lastData Then
    For r = 1 To UBound(m_fd(t), 1) - 4     '36 elements
        tfdei(r) = m_fd(t)(r, 10)
    Next

    'Leontief equation
    v_gobp(t) = MMult(MInv(MSub(m_i, m_ticn(t))), tfdei)
```

Figure 24: **Code snippet from the subprogram *Calculate* I**

Both output *v_gobp(t)(r)* and intermediate consumption *v_ticnpp(t)(r)* by industries calculated from the IO framework need to be transferred to the economic activities (*v_gocp(t)(r) and v_tiicp(t)(r)*) which corresponds to the STAN industrial data (ISIC rev. 4) published by OECD. The supply table provides a detailed overview of transactions in goods and services by industries which is used as transaction table *m_stpi(t)(r,c)*. Value added *v_vacp(t)(r)* is simply the difference between the previously mentioned variables.

The value added components compensation of employees *v_coe(t)(r)*, gross operating surplus and mixed income *v_gosmi(t)(r)* as well as other taxes less subsidies on production *v_otlsp(t)(r)* are determined in the modelling context. It is assumed that the latter variables changes accordingly with production *v_gocp(t)(r).* Compensation of employees depends on the number of employees *v_empl(t)(r)* and wages per capita *v_wpc(t)(r)* (Figure 25). Besides from intermediate inputs *v_tiicp(t)(r)*, compensation of employees and other taxes less subsidies on production feed back into the calculation of unit costs.

Figure 25 shows the calculation of the employees *v_empl(t)(r)* and compensation of employees *v_coe(t)(r)* for all industries. Since these variables are vectors, the calculation can be done in three lines instead of repeating the calculation for all 38 industries (red rectangles in Figure 25). *r* indicates the number of vector elements which is derived automatically from the row specification in the worksheet *Dataset*. The VBA function *UBound* returns the maximum size of the array (here: 38 industries). With the *for next* loop statement the equation is executed 38 times. The *UBound* function provides high flexibility: the function returns the upper bound and thus the number of elements in the array so that if the number of vector elements is changed in the *Dataset*, there is no need for adopting the code.



```
'(4.d) Employment
If [illegible]
    For r = 1 To UBound(v_empl(t), 1)
        v_empl(t)(r) = v_lc(v_lc_lastData)(r) * v_gov(t)(r) / 1000
    Next
    s_empl(t) = VecSum(v_empl(t))
End If

'(4.e) Compensation of employees
If [illegible]
    For r = 1 To UBound(v_coe(t), 1)
        v_coe(t)(r) = v_empl(t)(r) * v_wpc(t)(r) / 1000
    Next
End If
```

Figure 25: **Code snippet from the subprogram *Calculate* II**

Gross operating surplus and mixed income is calculated as a residual from value added and the remaining components.

The labour market is modelled at an aggregate level. Demographic development, i. e. population at working age *v_pag(t)(4)* determines labour force *s_lfce(t)*. Employment at industry level *v_empl(t)(r)* is explained by historically observed labour coefficients *v_lc(t)(r)* and price adjusted output *v_gov(t)(r)*. A wage rate per capita over all industries *s_wpc(t)* is calculated as a function which forecasts the result of the bargaining process between the unions and the firms: Macroeconomic labour productivity *v_gdpev(t)(1) / s_empl(t)* and the consumer price index *v_gdped(t)(2)* determines the macro wage rate *s_wpc(t)*, which, in turn, drives the wage rates per capita for all industries *v_wpc(t)(r)*.

The system of national accounts and balancing items (SNAB) is an integral part of the modelling system which allows for modelling the complete economic circuit and the monetary flows from production to consumption as well as the interaction of economic actors. Important variables are derived within the SNAB are, for example, the disposable income and net lending/borrowing.

Many variables from the IO framework such as value added, final consumption expenditures, gross fixed capital formation etc. serves as input into the SNAB. Other SNAB variables such as social contributions and savings are determined within the system either by estimating them or by definition.

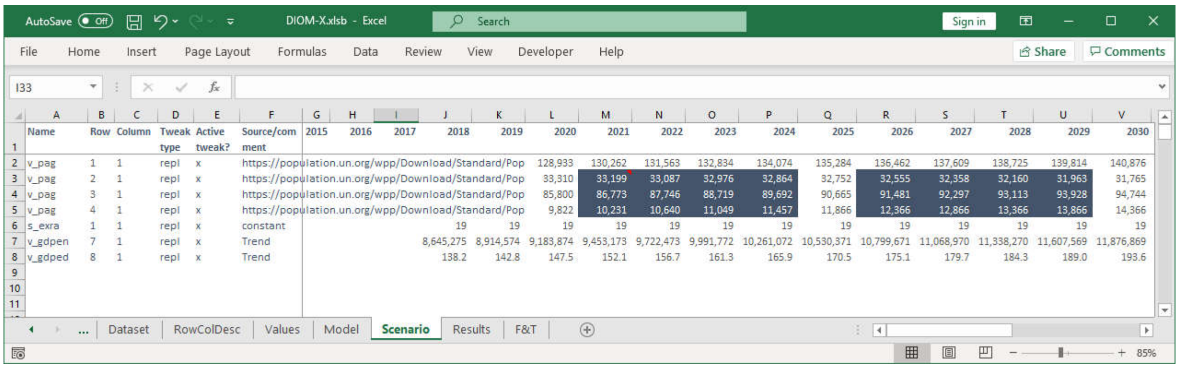
3.4.5 SCENARIO ANALYSIS

Once the model building process is finished and the model executes flawlessly, the most basic forecast is done. This forecast is both based on the premises that past behaviour is also effective in the future (business as usual scenario, BAU) as well as on a few exogenous assumptions. While the first premise is part of the regression analysis, the latter is specified in the *Scenario* worksheet (Figure 26). The model framework reads the scenario settings ("tweaks") from the *Scenario* worksheet at model runtime and injects these settings by changing the (calculated) variable values to the values given in the worksheet. So far, all variables which are given in this worksheet are exogenous variables meaning they are not dependent on other model variables, e. g. population *v_pag*, nominal exports *v_gdpen(t)(7)*. While the population forecast by age groups is given by the World populations prospects from the United Nations, nominal exports follow a trend. The applied tweak type is *repl*.

DIOM-X provides four types of tweaks:

- *repl* *replace current value*
- *mult* *multiply current value by given value*
- *gr* *apply given growth rate in % to value of previous year*
- *add* *add given value to current value*

If a tweak is to be activated, a "*x*" must be placed in column D. Otherwise the tweak is not active.

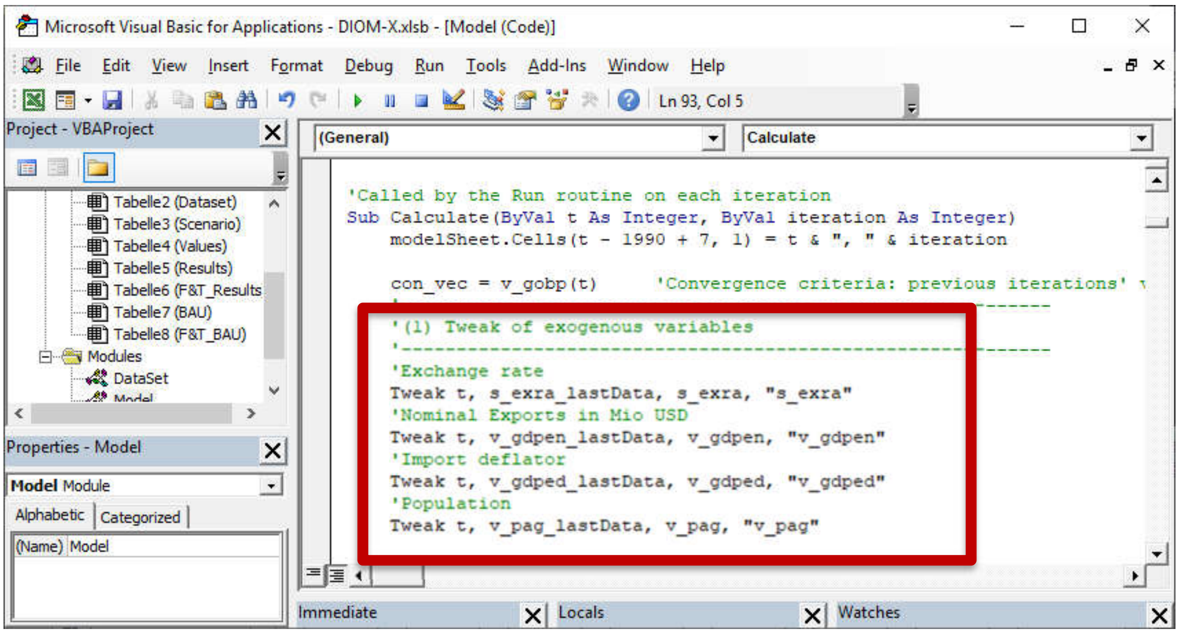


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	Name	Row	Column	Tweak type	Active tweak?	Source/comment	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030
2	v_pag	1	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						128,933	130,262	131,563	132,834	134,074	135,284	136,462	137,609	138,725	139,814	140,876
3	v_pag	2	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						33,310	33,199	33,087	32,976	32,864	32,752	32,555	32,358	32,160	31,963	31,765
4	v_pag	3	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						85,800	86,773	87,746	88,719	89,692	90,665	91,481	92,297	93,113	93,928	94,744
5	v_pag	4	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						9,822	10,231	10,640	11,049	11,457	11,866	12,366	12,866	13,366	13,866	14,366
6	s_exra	1	1	repl	x	constant				19	19	19	19	19	19	19	19	19	19	19	19	19
7	v_gdpen	7	1	repl	x	Trend				8,645,275	8,914,574	9,183,874	9,453,173	9,722,473	9,991,772	10,261,072	10,530,371	10,799,671	11,068,970	11,338,270	11,607,569	11,876,869
8	v_gdped	8	1	repl	x	Trend				138.2	142.8	147.5	152.1	156.7	161.3	165.9	170.5	175.1	179.7	184.3	189.0	193.6
9																						
10																						
11																						

Figure 26: **Worksheet *Scenario***

The tweaks must be implemented at an appropriate position in the VBA code as well (Figure 27). For the exogenous variables, it makes sense to apply these tweaks at the beginning of the *Calculate* subprogram in order to make them available to subsequent calculations.

The *tweak* function makes sure that tweaks can only be applied if the current year *t* is greater than *lastdata* before tweaks are applied. This protects historical data from being accidently overwritten.



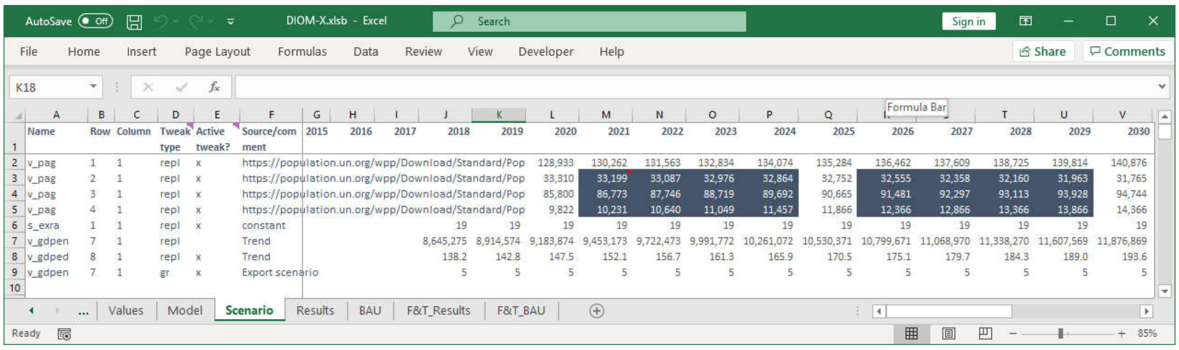
```
'Called by the Run routine on each iteration
Sub Calculate(ByVal t As Integer, ByVal iteration As Integer)
    modelSheet.Cells(t - 1990 + 7, 1) = t & ", " & iteration

    con_vec = v_gobp(t)      'Convergence criteria: previous iterations' v
    '-----------------------------------------------------------------------
    '(1) Tweak of exogenous variables
    '-----------------------------------------------------------------------
    'Exchange rate
    Tweak t, s_exra_lastData, s_exra, "s_exra"
    'Nominal Exports in Mio USD
    Tweak t, v_gdpen_lastData, v_gdpen, "v_gdpen"
    'Import deflator
    Tweak t, v_gdped_lastData, v_gdped, "v_gdped"
    'Population
    Tweak t, v_pag_lastData, v_pag, "v_pag"
```

Figure 27: Code snippet from the subprogram *Calculate* III

Once the BAU scenario is finalized the *Result* worksheet should be copied and renamed e. g. BAU. Thus, when carrying out scenario analysis the results of the BAU will not be overwritten. Each time when do the model run again, new results are stored in the *Results* worksheet.

Scenario or impact analysis is the main purpose of DIOM-X. Impacts of policy measures or adjustments of certain model assumptions can be analysed before they are implemented as actions, laws or regulations by defining a set of consistent assumptions and feeding them into the model.

For example, the exports which are following a trend in the BAU scenario are assumed to grow at a rate of 5 % (Figure 28).



Name	Row	Column	Tweak type	Active tweak?	Source/comment	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030
v_pag	1	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						128,933	130,262	131,563	132,834	134,074	135,284	136,462	137,609	138,725	139,814	140,876
v_pag	2	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						33,310	33,199	33,087	32,976	32,864	32,752	32,555	32,358	32,160	31,963	31,765
v_pag	3	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						85,800	86,773	87,746	88,719	89,692	90,665	91,481	92,297	93,113	93,928	94,744
v_pag	4	1	repl	x	https://population.un.org/wpp/Download/Standard/Pop						9,822	10,231	10,640	11,049	11,457	11,866	12,366	12,866	13,366	13,866	14,366
s_exra	1	1	repl	x	constant				19	19	19	19	19	19	19	19	19	19	19	19	19
v_gdpen	7	1	repl		Trend				8,645,275	8,914,574	9,183,874	9,453,173	9,722,473	9,991,772	10,261,072	10,530,371	10,799,671	11,068,970	11,338,270	11,607,569	11,876,869
v_gdped	8	1	repl	x	Trend				138.2	142.8	147.5	152.1	156.7	161.3	165.9	170.5	175.1	179.7	184.3	189.0	193.6
v_gdpen	7	1	gr	x	Export scenario				5	5	5	5	5	5	5	5	5	5	5	5	5

Figure 28: Example scenario: Export promotion

After the model run, the results are stored in the *Results* worksheet. The worksheets *AbsDiffResultsBAU* and *RelDiffResultsBAU* compare the results of both scenarios as absolute deviations in given units respectively in relative deviations in percentage. Further evaluation sheets with figures and tables can be created by the user, which show for example selected results in graphs or tables.

The comparative presentation of selected results of the export and BAU scenario shows the expected results: Higher nominal export yield in a higher GDP although imports are increasing as well. Prices are increasing – except import prices – due to higher costs especially labor costs which are related to a higher employment. More complex scenarios may yield results which are not self-evident at the beginning but prove to be correct by examining the feedback effects.

3.5 CONCLUSIONS

Most model builders use Excel either for simple (static) models only or as a pre/post data processing tool with more complex models. DIOM-X was built to examine whether Excel and especially the integrated VBA programming language are both suitable and sufficient to build elaborated dynamic input-output models. First results show, that this approach has some advantages especially for beginners in model building:

1. The software package offers most features that are needed out-of-the box. Some minor missing features such as matrix/vector algebra can be implemented without too much effort.
2. Execution speed is more than sufficient with small to medium sized models.
3. With the self-contained DIOM-X package, model distribution and sharing is very easy because the only technical prerequisite is a computer with preinstall Excel.
4. Potential model builders have to get acquainted to just one software package/programming language only and can reuse their knowledge with spreadsheet programs. Depending on the model size, a more sophisticated regression tool might be necessary.
5. The integrated debugger helps with model misspecifications and bugs because all data can be inspected at runtime.
6. The DIOM-X package is comparably easy to learn, esp. because of the clear structure, simple but powerful VBA language and thus seems to be a suitable toolset for capacity building.
7. By providing the full source code for both the model framework and the model template, the DIOM-X package can be adopted or enhanced to meet given specifications.

Currently, there is no straightforward way to get regression statements into the model. Possible options are to either enhance the DIOM-X framework by a implementing a regression tool in VBA which uses the data from the *Values* worksheet or to provide import and/or export routines from/for common statistics software packages. One example is to convert the contents of the *Values* worksheet to a CSV (comma separated values) file where the variables are stored in columns and the values per year stored in rows.

4 SUMMARY AND OUTLOOK

A static IO model can be easily implemented in any spreadsheet program and a great number of examples can be found in books and on the internet which help with the implementation.

Dynamic IO models are much more complicated so that their implementation is usually time-consuming, esp. because there are not many complete exercises that a potential model builder can start with.

The DIOM-X model building framework tries to fill in the gap by providing both a technical as well as a practical solution. Since all the information (system code, data, template model) is contained in one workbook, the model can be easily set up and distribution/sharing is just a matter of transferring that file to possible recipients. MS Excel is already widely used for data preparation as well as data evaluation, thus the next logical step seems to be to also perform the calculations within this software. Both the features and the performance are more than sufficient to implement a dynamic IO model.

Many potential users are skeptical when it comes to using computer models to address economic issues, partly because they often have no access to the model and/or model code. In contrast, a DIOM-X model is not a "black box". The framework contains the full source code for its data processing, model execution and scenario analysis statements. Thus, the model can be fully verified, tailored and/or extended to meet specific requirements.

The only technical prerequisite of implementing a DIOM-X is a computer with MS Excel/Office installed. In combination with the integrated model template, DIOM-X becomes a much easier to use the environment for capacity building in IO model theory and application.

DIOM-X has still a lot of room for improvement. The initial version needs a much better documentation if the system is to be used for capacity building. The calculation engine could be improved as well, i. e. by providing an interface for regression analysis and by simplifying the system code. Furthermore, the model template could be revised to even better meet the requirements of new model builders. With more models built in DIOM-X, comparing results and improving the models becomes much easier.

REFERENCES

Almon, C., 1991. The INFORUM Approach to Interindustry Modeling. Economic System Research 3/1, 1-7

Almon, C. (2014): The Craft of economic modelling – Part 1. Fifth edition. January 2014. Department of Economics, University of Maryland.

Ahlert, G., Distelkamp, M., Lutz, C., Meyer, B., Mönnig, A. & Wolter, M. I. (2009): Das IAB/INFORGE-Modell. In: Schnur, P. & Zika, G. [Hrsg]: Das IAB/INFORGE-Modell. Ein sektorales makroökonometrisches Projektions- und Simulationsmodell zur Vorausschätzung des längerfristigen Arbeitskräftebedarfs. IAB-Bibliothek 318, Nürnberg, S. 15-175.

BEA (n. d.): RIMSII. An essential tool for regional developers and planners. Bureau of Economic Analysis. U.S: Department of Commerce.

Cambridge Econometrics (2014): E3ME. Technical Manual, Version 6.0, Cambridge.

Eurostat (2008): Eurostat manual of Supply, Use and Input-Output Tables. Luxembourg. https://ec.europa.eu/eurostat/documents/3859598/5902113/KS-RA-07-013-EN.PDF/b0b3d71e-3930-4442-94be-70b36cea9b39

Gretton, P. (2013): On input-output tables: used and abuses. Productivity Commission Staff Research Note.

Großmann, A. & Lutz, C. (2017): Economic effects of reforming energy tax exemptions for the industry in Germany. In: Weishaar, S. E., Kreiser, L., Milne, J. E., Ashiabor, H. & Mehling, M. (eds.): The Green Market Transition. Carbon Taxes, Energy Subsidies and Smart Instrument Mixes. Critical Issues in Environmental Taxation, Vol. XIX, pp. 128-142, Edward Elgar. http://dx.doi.org/10.4337/9781788111171

Großmann, A., Lehr, U., Lutz, C., Mönnig, A. & Kleissl, S. (2016): Planning Policy Impact Assessments and Choosing the Right Methods: Manual for Development Practitioners. Published by GIZ, Eschborn. https://www.giz.de/fachexpertise/downloads/giz2016-en-Planning_Policy_Impact_Assessments.pdf

Großmann, A. & Hohmann, F. (2016): E3.pt – An E3 PortableDyme Model for Portugal. Proceedings (Vol. 1) of the 8th International Congress on Environmental Modelling and Software for supporting a sustainable future (iEMSs 2016) in Toulouse (France).

Großmann, A. & Hohmann, F. (2015): PortableDyme – A simplified software package for model building. Proceedings of the 23rd conference of the International Input Output Association in Mexico City (Mexico).

Großmann, A., Hohmann, F. & Wiebe, K. S. (2013): PortableDyme – a simplified software package for econometric model building. In: Bardazzi, Gezzi (Ed.): Macroeconomic modelling for policy analysis. Firenze University Press.

Holub, H.-W., Schnabl, H. (1994): Input-Output-Rechnung: Input-Output-Analyse. Oldenbourg Verlag, München.

ILO (2017): How to measure and model social and employment outcomes of climate and sustainable development policies. Training guidebook. Green Jobs Assessment Institution Network (GAIN), Geneva.

Lehr, U., Mönnig, A., Missaoui, R. & Marrouki, S. (2012): Renewable energy and energy efficiency in Tunisia – employment, qualification and economic effects , Study comissioned by Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ), Tunis.

Meade, D.S. (2014): Some Thoughts about the Interindustry Macroeconomic Model. Paper presented at the 22nd International Input-Output Conference July 14 – 17, 2014 Lisbon.

Nazara, S., Guo, G., Hewings, G. J. D. & Dridi, C. (2003): PyIO: Input-Output Analysis with Python. Regional Economics Applications Laboratory, University of Illinois at Urbana-Champaign.

Stocker, A., Großmann, A., Madlener, R. & Wolter, M. I. (2011): Sustainable energy development in Austria until 2020: Insights from applying the integrated model "e3.at". Energy Policy, 39 (10), pp. 6082-6099, doi: 10.1016/j.enpol.2011.07.2009

Kratena, K., Streicher, G., Temurshoev, U., Amores, A. F., Arto, I., Mongelli, I., Neuwahl, F., Rueda Cantuche, J. M., Andreoni, V., 2013. FIDELIO 1: Fully Interregional Dynamic Econometric Long-term Input-Output Model for the EU27. JRC Scientific and Policy Reports, European Commission, Luxemburg.

Leontief, W., 1986. Input-Output Economics. 2nd ed., New York, Oxford University Press.

Lewney, R. , Pollitt, H. & Mercure, J.-F. (2019): From input-output to macro-econometric model. Paper presented to the 27th International Input-Output Association Conference, 30th June to 5th July 2019, Glasgow, Scotland https://www.iioa.org/conferences/27th/papers/files/3602_20190429051_Frominput-outputtomacro-econometricmodel-Lewneyv1.0.pdf

Meade, D.S., 2014. Some Thoughts about the Interindustry Macroeconomic Model. Paper presented at the 22nd International Input-Output Conference July 14 – 17, 2014 Lisbon.

Meyer, B., Ahlert, G., 2016. Imperfect Markets and the Properties of Macro-Economic-Environmental Models as Tools for Policy Evaluation. GWS Discussion Paper 2016/9, Osnabrück.

Miller, R. E., Blair, P.D. (2009): Input-Output-Analysis. Foundations and Extensions. Cambridge University Press. Second Edition, New York.

Oosterhaven, J. (1996): Leontief versus Ghoshian Price and Quantity Models. Southern Economic Journal Vol. 62, No. 3 (Jan., 1996), pp. 750-759. DOI: 10.2307/1060892

Thijs, t. R. (2017): Handbook of Input-Output Analysis. Edward Elgar, Cheltenham, UK, Northampton, USA.

West, G.R., 1995. Comparison Input-Output, Input-Output + Econometric and Computable General Equilibrium Impact Models at the Regional Level. Economic System Research, Vol. 7, No. 2, 209-227.